Deutsche Ausgabe

LINUX

IN A NUTSHELL

Jessica Perry Hekman und die
Mitarbeiter von O'Reilly & Associates

O'REILLY™

Deutsche Übersetzung von Matthias Kalle Dalheimer

Linux in a Nutshell

Jessica Perry Hekman und die
Mitarbeiter von O'Reilly & Associates

Deutsche Übersetzung von
Matthias Kalle Dalheimer

O'REILLY™

Cambridge · Köln · Paris · Sebastopol · Tokyo

Die Information in diesem Buch wurde mit größter Sorgfalt erarbeitet. Dennoch können Fehler nicht vollständig ausgeschlossen werden. Verlag, Autoren und Übersetzer übernehmen keine juristische Verantwortung oder irgendeine Haftung für eventuell verbliebene fehlerhafte Angaben und deren Folgen.
Alle Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt und sind möglicherweise eingetragene Warenzeichen. Der Verlag richtet sich im wesentlichen nach den Schreibweisen der Hersteller. Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Alle Rechte vorbehalten einschließlich der Vervielfältigung, Übersetzung, Mikroverfilmung sowie Einspeicherung und Verarbeitung in elektronischen Systemen.

Kommentare und Fragen können Sie gerne an uns richten:
O'Reilly Verlag
Balthasarstraße 81
50670 Köln
Tel.: 0221/9731600
Fax: 0221/97316015
E-Mail: kommentar@oreilly.de

Copyright der deutschen Ausgabe:
© 1997 by O'Reilly Verlag
1. Auflage 1997
1. Nachdruck Januar1998

Die Originalausgabe erschien 1997 unter dem Titel
Linux in a Nutshell im Verlag O'Reilly & Associates, Inc.

Die Deutsche Bibliothek - CIP - Einheitsaufnahme

Hekman, Jessica Perry:
LINUX in a Nutshell ; Jessica Perry Hekman . [Übers. und dt. Bearb.: Matthias Kalle Dalheimer]. - 1. Aufl. - Köln : O'Reilly, 1997
Engl. Ausg. u.d.T.: Hekman, Jessica Perry : LINUX in a nutshell
ISBN 3-930673-57-6

Übersetzung: Matthias Kalle Dalheimer
Lektorat: Michael Gerth, Köln
Satz: Christoph Badura, Köln
Umschlaggestaltung: Edie Freedman & Hanna Dyer, Boston
Produktion: TYP*isch* Müller, München
Belichtung, Druck und buchbinderische Verarbeitung: Druckerei Kösel, Kempten

ISBN 3-930673-57-6

Dieses Buch ist auf 100% chlorfrei gebleichtem Papier gedruckt.

Inhaltsverzeichnis

Vorwort ..... *ix*

Kapitel 1 — Linux-Benutzerbefehle *1*

Kapitel 2 — Die UNIX Shells: Ein Überblick *123*

Sinn und Zweck einer Shell 123
Arten von Shells 124
Gemeinsamkeiten 125
Unterschiede 126

Kapitel 3 — bash: Die Bourne Again Shell *127*

Funktionsübersicht 127
Aufruf der Shell 128
Syntax 129
Variablen 137
Arithmetische Ausdrücke 140
Befehls-History 141
Eingebaute Befehle 143
Job-Verwaltung 160

Kapitel 4 — csh und tcsh *163*

Funktionsübersicht 163
Aufrufen der Shell 164

Syntax 165
Variablen 168
Ausdrücke 177
Befehls-History 179
Manipulation der Kommandozeile 182
Eingebaute csh- und tcsh-Befehle 186
Job-Verwaltung 205

Kapitel 5 — Mustervergleiche *207*

Dateinamen und Muster 207
Metazeichen in Linux-Programmen 208
Metazeichen 209
Beispiele zum Suchen 210

Kapitel 6 — Der Editor Emacs *213*

Einführung 213
Typische Probleme 213
Befehlsübersicht nach Gruppen sortiert 215
Befehlszusammenfassung nach Tastenkombinationen 220
Befehlsübersicht nach Befehlsnamen sortiert 223

Kapitel 7 — Der Editor vi *229*

Überblick über das Arbeiten mit dem vi 229
Cursorbewegungen 232
Editierbefehle 233
Speichern und Beenden 235
Bearbeiten mehrerer Dateien 235
Shell-Interaktion 236
Makros 236
Diverse Befehle 236
Alphabetische Tastenliste 237
vi-Einstellungen 239

Kapitel 8 — Der Editor ex *243*

Syntax der ex-Befehle 243
Alphabetisch sortierte Übersicht der ex-Befehle 245

Kapitel 9 — Der Editor sed ... *255*

Überblick über **sed** ... 255

Kommandozeilensyntax ... 256

Syntax der **sed**-Befehle ... 256

Nach Gruppen sortierte Zusammenfassung der sed-Befehle ... 257

Alphabetisch sortierte Übersicht der sed-Befehle ... 258

Kapitel 10 — Die Skriptsprache gawk ... *267*

Überblick über gawk ... 267

Kommandozeilensyntax ... 268

Muster und Prozeduren ... 269

gawk-Systemvariablen ... 271

Operatoren ... 272

Variablen- und Feldzuweisungen ... 272

Nach Gruppen sortierte Aufstellung der gawk-Befehle ... 273

Alphabetisch sortierte Zusammenfassung der Befehle ... 273

Kapitel 11 — Perl-Schnellübersicht ... *279*

Kapitel 12 — Überblick über die Befehle zum Programmieren ... *309*

Gebräuchliche Befehle ... 309

Alphabetische Befehlsübersicht ... 310

Kapitel 13 — Überblick über System- und Netzwerkverwaltung . *347*

Häufig benutzte Befehle ... 347

Netzwerk-Überblick ... 351

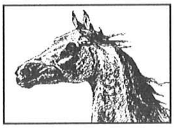
Überblick über TCP/IP ... 353

Überblick über NFS ... 358

Kapitel 14 — Befehle zur System- und Netzwerkverwaltung ... *361*

Alphabetische Zusammenfassung der Befehle ... 361

Index ... *449*



Vorwort

Dies ist ein Buch über Linux, eine frei erhältliche Variante des Betriebssystems UNIX für Personal Computer. Zusätzlich zum Code des Betriebssystemkerns enthält Linux Werkzeuge und Befehle aus dem GNU-Projekt der Free Software Foundation (FSF) und aus Berkeley UNIX (BSD) und eine vollständige Portierung des X Window Systems (XFree86) des X Consortium, außerdem natürlich viele Fähigkeiten, die speziell für Linux entwickelt wurden.

Dieses Buch ist eine Referenz zum Nachschlagen der wichtigsten Befehle und Funktionalitäten des Betriebssystems Linux. Wie die anderen Bücher der „in a Nutshell"-Reihe von O'Reilly richtet sich dieses Buch an Benutzer, die wissen, was sie erreichen möchten, und eine ungefähre Vorstellung davon haben, wie sie dies erreichen können, aber denen gerade nicht der richtige Befehl oder der richtige Parameter einfällt. Wir hoffen, daß dieses Buch ein wertvolles Nachschlagewerk auf dem Schreibtisch der Linux-Benutzer wird.

Das Aufregende an Linux

Mit Linux leben die große Kreativität und das Prinzip des Teilens wieder auf, für das UNIX lange Zeit bekannt war. Viele erinnern sich wehmütig an die Zeit, als AT&T UNIX umsonst an Universitäten weitergab und die University of Berkeley begann, ihre Version so zu verteilen, daß jeder an sie herankommen konnte. Für diese Veteranen kann Linux den Geist der Zusammenarbeit zurückbringen, umso mehr, als das Internet heute weit verbreitet ist. Und für die vielen Menschen, die zu jung sind, um sich an diese ersten offenen Systeme zu erinnern (so wie die vielen Studenten, die Linux-Anhänger sind), oder deren bisherige Erfahrungen schmerzhaft durch proprietäre Betriebssysteme eingeschränkt wurden, ist jetzt die Zeit gekommen, das Wunder frei verteilbaren Quelltextes und beliebig anpaßbarer Schnittstellen zu entdecken.

Der Linux-Kernel wurde ursprünglich von Linus Torvalds an der Universität Helsinki in Finnland entworfen und später von ihm in Zusammenarbeit mit vielen Freiwilligen aus aller Welt weiterentwickelt. Mit „Kernel" meinen wir den Kern des Betriebssystems selber — nicht die Applikationen (wie Compiler, Befehlsinterpreter usw.), die darauf laufen. Heute sind mit dem

Begriff „Linux" oft der Kernel, die Applikationen und die gesamte Systemumgebung zusammen gemeint.

Linux-Systeme dürfen genaugenommen nicht als „UNIX-Version" bezeichnet werden, weil sie nicht den offiziellen Tests unterworfen und entsprechend lizenziert worden sind.* Linux bietet aber alle von Standard-UNIX-Systemen bekannten Programmierschnittstellen. Die meisten UNIX-Werkzeuge wurden auf Linux portiert, wie Sie aus diesem Buch ersehen können. Linux ist ein mächtiges, robustes, voll verwendbares System für Leute, die UNIX mögen.

Anfänger finden die Flexibilität und Offenheit von UNIX gewöhnlich verwirrend und frustrierend, aber wenn sie sich einmal daran gewöhnt haben, fragen sie sich, wie sie ohne es auskommen konnten. Diese Flexibilität und Offenheit hat zu vielen Erweiterungen geführt, z.B. zu neuen Werkzeugen wie Perl und Experimenten im Bereich der Informatik, die in einer Reihe von Fällen in kommerziellen Computer-Systemen aufgegangen sind.

Ein Grund für den Erfolg von Linux besteht darin, daß Linux eine gewaltige Menge von Hardware für IBM-kompatible Personal Computer unterstützt. Leute, die an MS-DOS und Microsoft Windows gewöhnt sind, sind oft erstaunt und begeistert, wieviel schneller ihre Hardware mit Linux arbeitet — Linux macht von den vorhandenen Ressourcen einen sehr effizienten Gebrauch. Inzwischen gibt es Linux auch für eine Reihe anderer Computer-Modelle, insbesondere den PowerPC und die Alpha-Reihe von Digital Equipment, eine der schnellsten Workstations überhaupt. Portierungen auf Sparc, m68k und andere befinden sich bereits in Arbeit.

Möglicherweise arbeiten Sie schon aus einem von vielen Gründen mit Linux. Vielleicht benötigen Sie beruflich für sich ein Betriebssystem für ein bestimmtes Projekt und sind dabei an ein Netzwerk aus UNIX-Rechnern angeschlossen — und weil Sie mit UNIX-Rechnern zusammenarbeiten müssen, wollen Sie etwas Ähnliches auf Ihrem eigenen Rechner. Vielleicht sind Sie ein Computer-Anfänger, der eine Aufgabe erfüllen möchte, für die UNIX traditionell gut geeignet ist, wie z.B. der Betrieb eines Web-Servers. Vielleicht haben Sie gerade einen PC gekauft und wollen Linux zu Hause benutzen, weil Sie auch auf der Arbeit UNIX verwenden. Oder Sie sind es einfach leid, daß Ihr Betriebssystem entscheidet, was gut für Sie ist, und Sie daran hindert, das zu tun, was Sie eigentlich tun möchten — dann sind Sie ein Fall für Linux.

Programme unter Linux

Linux-Programme sind nicht dieselben wie auf Standard-UNIX-Systemen: Sie sind besser! Das liegt daran, daß die meisten aus dem GNU-Projekt der Free Software Foundation (FSF) stammen. Aus langjähriger Erfahrung mit Standard-UNIX-Hilfsprogrammen und dank Fortschritten im Bereich der Informatik haben die Programmierer im GNU-Projekt es verstanden, neue Versionen der Standard-Werkzeuge zu schaffen, die mehr Funktionen haben, schneller und effizienter arbeiten und bei denen viele der Fehler und Inkonsistenzen der Originale ausgemerzt sind.

Während das GNU-Projekt die Programmierwerkzeuge und Standard-Befehle wie *grep* zur Verfügung gestellt hat, stammen die meisten Werkzeuge zur System- und Netzwerk-Verwaltung aus der Berkeley Software Distribution (BSD). Zusätzlich wurden Werkzeuge speziell für Linux geschrieben, um Bereiche abzudecken, die es nur unter Linux gibt (z.B. bestimmte Dateisysteme). Dieses Buch dokumentiert all die Standard-Unix-Befehle, die gewöhnlich auf Linux-Distributionen verfügbar sind.

*Bevor ein Betriebssystem UNIX genannt werden darf, muß es sich einer Prüfung durch X/Open unterziehen.

Die dritte große Quelle von Linux-Software ist das X Consortium, das das X Window System entwickelte, und das XFree86-Projekt, das die ursprüngliche Portierung auf Intel-basierende Rechner von Thomas Roell und einigen anderen weiterentwickelte und ausbaute. Das X Window System ist nicht Thema dieses Buches; O'Reilly bietet aber andere Bücher hierzu an.

Quelltexte und Lizenzen

Sie erhalten Linux immer zusammen mit dem Quelltext. Dies gilt auch für alle Werkzeuge und Hilfsprogramme (es sei denn, Ihr Distributions-Hersteller bietet eine kommerzielle Applikation oder Bibliothek als besondere Dreingabe an). Vielleicht werden Sie nie einen Blick auf den Quelltext werfen, aber er ist der Schlüssel zur Stärke von Linux. Gemäß der Linux-Lizenz muß jeder, der Linux vertreibt, den Quelltext mitliefern. Das ermöglicht es denjenigen, die kompetent genug dazu sind, Fehler zu beheben, die Funktionsweise des Systems zu erlernen und dieses Wissen weiterzugeben und Verbesserungen zu machen, von denen wir alle profitieren. Die Linux-Lizenz ist die bekannte General Public License, auch als GPL oder „copyleft" bekannt, die von der FSF erdacht und populär gemacht wurde. Die FSF, gegründet von Richard Stallman, ist ein Phänomen, das viele Leute für unmöglich halten würden, wenn es nicht existierte. (Das gleiche gilt auch für Linux — wer hätte vor fünf Jahren daran geglaubt, daß ein robustes Betriebssystem in Gemeinschaftsarbeit über das Internet entwickelt und frei weitergegeben wird?) Einer der beliebtesten Editoren unter UNIX, GNU Emacs, kommt von der FSF, genau wie der *gcc* und der *g++* (C und C++ Compiler), die eine Zeitlang das Maß aller Dinge bildeten, was Optimierung und schnellen Code angeht.

Da sie sich der Weitergabe von Software gewidmet hat, stellt die FSF all ihren Code und die Dokumentation im Internet zur Verfügung und erlaubt es jedem, die Software zu verändern. Sie arbeitet an einem Projekt, das auf den ersten Blick Linux ähnelt, aber breiter angelegt ist: eine vollständige UNIX-Version, die frei verfügbar ist. Das ganze Projekt heißt GNU (was wiederum für *„GNU is not UNIX"*, GNU ist nicht UNIX, steht, dem „G" kann keine weitere Bedeutung zugeordnet werden) und das in der Entwicklung befindliche Betriebssystem HURD.

Um proprietäre Weiterentwicklungen auf der Basis von GNU-Code zu verhindern, verlangt die FSF, daß alle Erweiterungen ihres Codes unter derselben GPL-Lizenz zur Verfügung gestellt werden müssen, also auch mit Quelltexten. Dies soll Einzelpersonen und Institutionen ermuntern, Verbesserungen zu machen und diese wiederum mit anderen zu teilen. Das einzige, was nicht erlaubt ist, ist, Verbesserungen zu machen und diese dann ohne Weitergabe des Quelltextes zu verkaufen, weil damit der Vorteil der freien Quelltext-Weitergabe für die FSF und die Benutzer dahin wäre.

Wie schon weiter oben erwähnt, kommen viele Linux-Werkzeuge nicht aus dem GNU-Projekt, sondern von BSD. Auch bei BSD handelt es sich um freie Software. Die Lizenzen unterscheiden sich deutlich, was Sie als Benutzer aber nicht interessieren muß. Es läuft im wesentlichen darauf hinaus, daß Sie bei der BSD-Lizenz fast alles machen dürfen, sofern Sie die ursprünglichen Entwickler erwähnen, während bei der GPL der Quelltext zur Verfügung gestellt werden muß.

Was Sie in diesem Buch finden werden

Vor vielen Jahren veröffentlichte der O'Reilly Verlag *UNIX in a Nutshell* als eines seiner ersten Bücher. Dieses Buch fand sich bald weltweit auf den Schreibtischen eifriger Benutzer von UNIX. Viele Leute verwenden *UNIX in a Nutshell* anstelle der vermutlich offizielleren und vollständigeren Handbücher auf ihrem System. Manche sagen, daß dies das Buch ist, das sie von allen ihren Computer-Büchern am häufigsten verwenden.

UNIX in a Nutshell ist kein UNIX-Lehrbuch — schließlich ist es nur als schnelle Referenz gedacht —, aber sowohl Anfänger wie auch erfahrene Anwender benutzen es mit großem Gewinn. Wenn Sie in etwa wissen, welchen Befehl Sie benötigen, sich aber nicht sicher sind, wie er funktioniert oder welche Optionen Sie verwenden sollten, um die gewünschte Ausgabe zu erreichen, ist dieses Buch die richtige Stelle zum Nachschlagen. Auch kann es Sie auf Optionen hinweisen, von denen Sie bisher nichts wußten, und so noch produktiveres Arbeiten ermöglichen.

Linux in a Nutshell ist eine überarbeitete, modernisierte und angepaßte Version von *UNIX in a Nutshell* speziell für Linux. Aber nicht nur das; wir haben ein Buch herausgegeben, das auch viele UNIX-Benutzer verwenden werden, weil mit diesem Buch die erste Referenz-Dokumentation der Werkzeuge vorliegt, die von der FSF für das GNU-Projekt geschrieben wurden. GNU-Werkzeuge sind auf vielen UNIX-Systemen beliebt, so daß Sie diese vielleicht auch verwenden, wenn Sie Linux nicht einsetzen.

Wie viele Computersysteme aus der Zeit, in der UNIX entstand, ist auch Linux weitgehend ein befehlsbasiertes System. Zwar gibt es in manchen Linux-Versionen einige wenige grafische Werkzeuge zur Systemverwaltung, aber die interessanteren Dinge, die man mit Linux tun kann, sind befehlsgesteuert. Wenn Sie sich erst einmal an die UNIX Befehls-Interpreter (Shell genannt) und einige besonders häufig gebrauchte Befehle gewöhnt haben, werden Sie feststellen, daß man damit sehr viel schneller arbeiten kann als mit einer grafischen Oberfläche. Außerdem läßt sich das System so viel besser an Ihre Bedürfnisse anpassen. Aus diesen Gründen haben wir in diesem Buch alle Befehle dokumentiert, die Sie unter Linux am häufigsten verwenden werden.

Weitere Nachschlagemöglichkeiten

Unser Ziel bei der Produktion dieses Buches war Ihre Bequemlichkeit, und das bedeutet auch, es klein zu halten. Es enthält sicherlich nicht alles, was Sie in Ihrer Online-Dokumentation finden können.

Jede Distribution unterscheidet sich etwas von den anderen, aber Sie werden bemerken, daß die Befehle, die wir hier dokumentieren, diejenigen sind, die Sie die meiste Zeit benutzen werden, und daß diese in allen Distributionen gleich arbeiten. Wir behandeln hier grundlegende Befehle, Programmierwerkzeuge, System- und Netzwerkverwaltung und vieles andere. Aber einige Bereiche sind so umfangreich, daß wir sie auslassen mußten. TeX (ein Werkzeug zur Textverarbeitung, das vor allem im wissenschaftlichen Bereich, aber auch von Linux-Benutzern im allgemeinen sehr viel eingesetzt wird) konnte nicht mit aufgenommen werden, genauso wenig wie Tcl/Tk (eine Skriptsprache mit zugehörigen grafischen Werkzeugen). Außerdem wird das X Window System in diesem Buch nicht behandelt. Die Behandlung dieser Themen würde dazu führen, daß das Buch aus der Bindung quellen würde; sie bedürfen außerdem einer detaillierteren Behandlung, als es hier möglich wäre. Als Informationsquelle zu diesen Themen empfehlen wir die folgenden O'Reilly-Bücher:

- *Making TeX Work* von Norm Walsh
- *Tcl/Tk Tools* von Mark Harrison
- Die X Window System-Reihe, insbesondere *Volume 3: The X Window System User's Guide* von Valerie Quercia und Tim O'Reilly

Dieses Buch enthält auch keine Information darüber, wie ein Linux-System installiert und gewartet wird. Diese Information finden Sie in *Linux — Wegweiser zur Installation und Konfiguration* von Matt Welsh und Lar Kaufmann. Informationen über den Netzwerkbetrieb finden Sie in *Linux — Wegweiser für Netzwerker* von Olaf Kirch. Einen Überblick über Sprachen, Bibliotheken und Werkzeuge zum Programmieren bietet Ihnen *Linux — Wegweiser zur Programmierung und Entwicklung* von Matthias Kalle Dalheimer. Zusätzlich zu den Linux-Büchern sind vermutlich auch viele der anderen UNIX- und X-Bücher von O'Reilly für Sie interessant. Weitere nützliche Bücher, um das nötige Hintergrundwissen über System- und Netzwerkverwaltung zu bekommen, sind *Unix System Administration* von Æleen Frisch und *TCP/IP Netzwerk Administration* von Craig Hunt.

Auch das Internet enthält sehr viel Information über Linux. Eine der besten Quellen ist das Linux Documentation Project unter *http://sunsite.unc.edu/mdw*. Mit der Bereitstellung deutschsprachiger Informationen befassen sich das Deutsche Linux HOWTO Projekt (DLHP), zu erreichen zur Zeit unter *http://www.tu-harburg.de/~semb2204/dlhp/DE-HOWTO.html* und das Projekt: Deutsche Manpages unter *http://www.Infodrom.North.DE/Linux/manpages-de/*. Online-Information über die GNU-Werkzeuge, die in diesem Buch behandelt werden, finden Sie unter *ftp://prep.ai.mit.edu/pub/gnu/* oder *http://www.delorie.com/gnu/*.

Wie dieses Buch organisiert ist

Nach einer umfassenden Referenz aller relevanten Benutzerbefehle unter Linux widmen wir uns den Shells und den wichtigsten Editoren und Skriptsprachen. Es folgt ein Überblick über die Befehle zum Programmieren und abschließend finden Sie noch eine ausführliche Referenz der Befehle zur System- und Netzwerkverwaltung.

Kapitel 1, *Linux-Benutzerbefehle*, listet nicht nur alle Befehle und deren Optionen, sondern erklärt mit Beispielen auch ihre Anwendung.

Kapitel 2, *Die UNIX Shells: Ein Überblick*, Kapitel 3, *bash: Die Bourne Again Shell* und Kapitel 4, *csh und tcsh* widmen sich den Shells. Eine Shell ist einfach nur ein Programm, das Befehle vom Benutzer entgegennimmt und ausführt. Verschiedene Shells verwenden manchmal eine etwas unterschiedliche Syntax, um das gleiche auszudrücken. Unter Linux sind **bash** und **tcsh** zwei beliebte Shells, die subtile Unterschiede aufweisen (eines der schönen Merkmale von Linux und anderen UNIX-Systemen ist, daß Sie vielfach eine große Auswahl haben, wobei jede Möglichkeit ihre eigenen Stärken und Schwächen aufweist). Sie werden die Kapitel über Shells vermutlich lesen wollen, nachdem Sie sich einige Zeit mit Linux beschäftigt haben, weil diese sich hauptsächlich mit mächtigen, fortgeschrittenen Features befassen, die man erst als erfahrener Benutzer benötigt.

Davor müssen Sie einige große und umfassende Werkzeuge erlernen, um wirklich mit Linux arbeiten zu können: einen Editor, einen Compiler, eine Skriptsprache usw.

Kapitel 5, *Mustervergleiche*, erklärt in aller Kürze das Wichtigste, das Sie über Pattern Matching wissen müssen.

Kapitel 6, *Der Editor Emacs*, Kapitel 7, *Der Editor vi*, und Kapitel 8, *Der Editor ex*, beschreiben die beiden bekanntesten Editoren unter Linux: Emacs und vi sowie den Zeileneditor ex, auf dem vi aufgebaut ist.

Kapitel 9, *Der Editor sed*, Kapitel 10, *Die Skriptsprache gawk*, und Kapitel 11, *Perl-Schnellübersicht*, dokumentieren die unter Linux populären Skriptsprachen sed, gawk und Perl.

Kapitel 12, *Überblick über die Befehle zum Programmieren*, beschreibt vorhandene Programmierwerkzeuge. Die UNIX-Programmierumgebung hat für einige andere Systeme Modell gestanden und belohnt das Erlernen der Werkzeuge mit großartiger Unterstützung beim Erstellen von Programmen, Revisions-Kontroll-Systemen u.v.m.

Kapitel 13, *Überblick über die System- und Netzverwaltung* und Kapitel 14, *Befehle zur System- und Netzverwaltung*, behandeln die etwas größeren Themen System- und Netzwerkverwaltung.

Typografische Konventionen

Dieses Referenzhandbuch benutzt die folgenden typografischen Konventionen:

Fettdruck	wird verwendet für Befehle, Programme und Optionen. Alles, was in Fettdruck gedruckt ist, sollte wörtlich eingegeben werden.
Kursiv	wird verwendet für Argumente, Optionen und Variablen, die durch die für den Benutzer gültigen Werte ersetzt werden sollen. Weiter wird kursive Schrift für Datei- und Verzeichnisnamen sowie Kommentare in Beispielen benutzt.
`Courier`	wird für den Inhalt von Dateien und die Ausgabe von Befehlen verwendet.
`Courier Fett`	wird in Beispielen und Tabellen verwendet und zeigt Befehle oder anderen Text, der vom Benutzer wörtlich eingegeben werden sollte.
`Courier Kursiv`	wird in Beispielen und Tabellen verwendet und bezeichnet Text, der vom Benutzer durch eigene Werte ersetzt werden sollte.
`%`, `$`	bezeichnen in einigen Beispielen die Eingabeaufforderung der C-Shell (`%`) bzw. der Bourne Shell (`$`).
`[ ]`	umgeben optionale Bestandteile in Syntaxbeschreibungen. (Die eckigen Klammern selber werden nie eingegeben.) Beachten Sie bitte, daß in vielen Befehlen das Argument [*Dateien*] vorkommt. Wenn ein Dateiname weggelassen wird, wird die Standardeingabe (d.h. normalerweise die Tastatur) angenommen. Beenden Sie Ihre Eingabe mit dem Dateiendezeichen.
EOF	bezeichnet das Dateiendekennzeichen (normalerweise **CTRL-D**).
|	wird in Syntaxbeschreibungen verwendet und bezeichnet mehrere Elemente, aus denen nur genau eines ausgewählt werden darf.
→	wird am Ende einer rechten Seite verwendet, um anzuzeigen, daß der aktuelle Eintrag auf der nächsten Seite fortgesetzt wird. Die Fortsetzung wird mit ← markiert.

Noch eine Bemerkung zur Syntax: Oft kann das Leerzeichen zwischen einer Option und ihrem Argument weggelassen werden. In anderen Fällen müssen die Leerzeichen (oder die Tatsache, daß sie gerade nicht vorhanden sind) genau befolgt werden. So kann es sein, daß **-w***n* anders interpretiert wird als **-w** *n*. Sie sollten die Leerzeichen in der Syntaxbeschreibung einer Option genau beachten.

Danksagungen

Ich möchte allen meinen Freunden bei O'Reilly für ihre Hilfe bei der Arbeit an diesem Buch danken — meinen Lektoren Andy Oram und Robert Denn; der Produktionslektorin Jane Ellin, Stephen Spainhour, der dieses Buch fertiggestellt hat, als ich es selbst wegen meiner Sehnenscheidenentzündung nicht konnte; Seth Maislin, der wie üblich einen hervorragenden Index zusammengestellt hat; David Futato, der frühe Produktionsarbeiten übernommen hat; Madeleine Newell und Kismet McDonough-Chan, die ebenfalls bei der Produktion mitgeholfen haben; Clairemarie Fisher O'Leary und Sheryl Avruch für ihre Qualitätssicherung; allen aus der Werkzeug-Gruppe (besonders Lenny Muellner, der die meisten Werkzeuge für dieses Buch geschrieben hat, aber auch Erik Ray und Ellen Siever) sowie Edie Freedman, die die Titelseite entworfen hat. Norm Walsh und Tanya Herlick, meine Vorgesetzten, waren so großzügig, mir genug Zeit für die Arbeit an diesem Projekt zur Verfügung zu stellen (auch wenn ich eigentlich die ganze Zeit für Norm arbeiten sollte). Ein besonderer Dank geht an Stephen Figgins im technischen Support von O'Reilly, der sehr viele Befehle aus diesem Buch getestet hat. Dank auch an Greg Hankins vom Linux Documentation Project, der dieses Buch aus technischer Sicht korrekturgelesen hat. Schließlich steht dieses Buch in tiefer Schuld bei all den anderen O'Reilly-Büchern, die ich häufig zu Rate gezogen habe (was ein höflicher Ausdruck für „abgeschrieben" ist). Dieses Buch steht damit auf einer soliden Basis.

Die Monate, während denen ich an Linux in a Nutshell schrieb, verschwimmen in meinen Erinnerungen, aber ich kann mich gut an die Leute erinnern, die mir über den Sommer geholfen haben: Dave Vernal, JD Paul, Paul Berger, Elizabeth Haynes, Kevin Lyda, Beil Laughlin, meine Mutter und mein Bruder. Adam, ich kannte dich damals noch nicht, aber ich würde gern so tun, als hättest du quasi einen rückwirkenden Einfluß. Und dann natürlich alles Liebe an meine Katze, die ihre Haare auf meinem Monitor hinterließ und mir Bescheid sagte, wenn ich zu lange gearbeitet hatte.

Vorwort des Übersetzers

Die „... in a Nutshell" Bücher von O'Reilly haben schon eine lange Tradition. Um so mehr freut es mich, daß es mit „Linux in a Nutshell" Familienzuwachs gibt und ich die Gelegenheit hatte, durch eine Übersetzung dieses Buch auch deutschen Lesern nahezubringen.

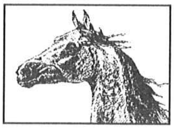
Die Übersetzung solcher Referenzhandbücher ist immer eine knifflige Angelegenheit, weil Sie gespickt sind mit angloamerikanischen Fachbegriffen. Übersetzt man diese konsequent, besteht die Gefahr, daß kein Leser mehr versteht, wovon die Rede ist. Übersetzt man sie gar nicht, hat man eigentlich keine Übersetzung, sondern eine Ansammlung englischer Substantive, die mit deutschen Verben und Füllwörtern verbunden werden. Gemeinsam mit dem O'Reilly-Lektorat haben wir daher versucht, einen Mittelweg zu finden. Ihnen als Leser obliegt es zu beurteilen, ob uns das gelungen ist.

Im Vergleich zur Originalausgabe wurden neben Fehlerkorrekturen folgende Änderungen durchgeführt:

- Vorwort und Kapitel 1 wurden im Vorwort zusammengefaßt.
- Kapitel 5 wurde um Anmerkungen zu Regulären Ausdrücken in Emacs Lisp und Perl ergänzt.
- Als Kapitel 11 wurde eine Schnellreferenz zu Perl 5 eingefügt.

Ich möchte mich besonders bei Sebastian Hetze bedanken, der das Kapitel 1 übersetzt hat, außerdem bei Dirk Hohndel für fachlichen und moralischen Beistand, beim O'Reilly-Team für die Zusammenarbeit und schließlich natürlich bei meiner Familie, die die hektische Phase vor der Drucklegung des Buches mit Gelassenheit und Karotten-Kartoffel-Brei überstanden hat.

Hamburg, im Mai 1997, Matthias Kalle Dalheimer



KAPITEL 1

Linux-Benutzerbefehle

In diesem Kapitel werden die Benutzerbefehle von Linux vorgestellt (im Unterschied zu den Befehlen zur Programmentwicklung und zur Systemadministration). Der Beginn jeder Befehlsbeschreibung ist durch den Befehlsnamen am äußeren Rand der Seite markiert. Auf die erste Zeile mit einer Kurzdarstellung der Kommandosyntax folgen eine kurze Beschreibung und eine Liste aller vorhandenen Optionen. Zu vielen Befehlen werden am Ende des Eintrags noch Beispiele gegeben. Wenn Sie nur eine schnelle Gedächtnisstütze oder eine Anregung für ein Kommando benötigen, können Sie gleich zu den Beispielen springen.

Eine Liste der typografischen Konventionen zur Beschreibung der Kommandos finden Sie im Vorwort. Zusätzliche Hilfestellung bei der Suche nach einem bestimmten Befehl finden Sie im Index am Ende dieses Buches.

Wenn Sie einen Befehl in diesem Kapitel nicht finden, ist er möglicherweise als Programmierbefehl oder als Administrationsbefehl aufgeführt. Sehen Sie in der alphabetischen Liste in den entsprechenden Kapiteln nach. Programme, die das X Window System benutzen, sind in diesem Buch nicht aufgeführt.

---

apropos

apropos [*Optionen*] *Suchwort*

Durchsucht die Manualpages nach dem Auftreten von *Suchwort*. Die Suche ist auf die Kurzbeschreibung am Beginn jeder Manualpage beschränkt.

Optionen

-m *System*[, ...], **--systems=***System*[, ...]
: Durchsucht auch die Manualpages auf einem entfernten *System*. Soll auf weiteren Systemen gesucht werden, separieren Sie die Systemnamen in der Liste durch Kommata.

-M *Pfad*, **--manpath=***Pfad*
: Sucht auf dem angegebenen *Pfad* statt auf dem normalen (in $MANPATH festgelegten) Verzeichnispfad nach den Manual-

→

apropos ←

pages. Die Verzeichnisse müssen durch Doppelpunkte separiert werden.

-r, --regex
: Ermöglicht die Auswertung von regulären Ausdrücken im *Suchwort*.

-w, --wildcard
: Jokerzeichen im *Suchwort* werden im Stil der Shell ausgewertet.

arch

arch

Gibt den Typ der Rechnerarchitektur aus.

at

at [*Optionen*] *Zeit*

Führt die auf der Standardeingabe eingegebenen Kommandos zu einer bestimmten *Zeit* und optional an einem bestimmten *Datum* aus. (Sehen Sie auch **batch**.) Die Eingabe wird durch *EOF* beendet. Die *Zeit* kann entweder als Zahlenwert für die Stunde (optional mit Minuten und Zusätzen) oder durch ein Schlüsselwort angegeben werden. Das Kommando kann ein Datum enthalten, das als Monat mit Tag, als Wochentag oder durch ein spezielles Schlüsselwort (*today* oder *tomorrow*) ausgedrückt wird. Ein Zuschlag kann ebenfalls angegeben werden. Einzelheiten werden in den folgenen Absätzen erläutert.

Optionen

-b
: Das Programm arbeitet wie **batch**.

-d
: Das Programm arbeitet wie **atrm**.

-f *Datei*
: Liest das Kommando aus der *Datei* anstelle der Standardeingabe.

-l [*Job_id*]
: Gibt alle Jobs für den aktuellen Benutzer oder für die Job-Queue aus, wenn eine *Job_id* angegeben wurde. (Das Programm arbeitet wie **atq**.)

-m
: Sendet eine Mail an den Benutzer, wenn der Job fertig ist, unabhängig davon, ob bei dem Job eine Ausgabe erzeugt wurde.

-q *Buchstabe*
: Reiht den Job in die mit *Buchstabe* bezeichnete Warteschlange ein, wobei die *Buchstaben* von a–z erlaubt sind. Die Standardwarteschlange ist c. (Die Warteschlange für batch ist standardmäßig b.) Je höher der Buchstabe für die Warteschlange ist, desto niedriger ist die Priorität, mit der die Jobs dieser Warteschlange laufen. Mit Großbuchstaben gekennzeichnete Warteschlangen werden an batch weitergegeben.

Zeit

hh:mm [*Zusatz*]

Stunden können eine oder zwei Ziffern haben (standardmäßig wird eine 24-Stunden-Uhr angenommen); zusätzliche Minuten können eine oder zwei Stellen haben; der Doppelpunkt kann entfallen, wenn die Zeit im Format *h*, *hh*, oder *hhmm* angegeben wird. Zum Beispiel sind 5, 5:30, 0530, 19:45 gültige Zeitangaben. Wenn der Zusatz **am** oder **pm** hinzugefügt wird, basiert die Zeitangabe auf einer 12-Stunden-Uhr. Wenn der Zusatz **zulu** angehängt wird, bezieht sich die Zeitangabe auf Greenwich Mean Time.

midnight | **noon** | **teatime** | **now**

Jedes dieser Schlüsselwörter kann anstelle einer numerischen Zeitangabe verwendet werden. **teatime** entspricht 4 Uhr nachmittags; **now** muß ein *Zuschlag* folgen.

Datum

Monat Zahl[, *Jahr*]

Monat ist der englische Name eines der zwölf Kalendermonate, ausgeschrieben oder auf die drei ersten Buchstaben abgekürzt; *Zahl* ist der Kalendermonat, *Jahr* ist die vierstellige Jahreszahl. Wenn der Monat vor dem aktuellen Kalendermonat liegt, wird der Job für das folgende Jahr einsortiert.

Tag Der englische Name eines der sieben Wochentage, ausgeschrieben oder auf die ersten drei Buchstaben abgekürzt.

today | **tomorrow**

Gibt den aktuellen oder den nächsten Tag an. Wenn das *date* weggelassen wird, reiht **at** einen Job für **today** am gleichen Tag ein, wenn die angegebene *Zeit* später eintritt, anderenfalls am folgenden Tag.

Zuschlag

Geben Sie einen numerischen Zuschlag an, wenn Sie die Ausführungszeit oder den Tag *relativ* zur aktuellen Zeit angeben möchten. Der Zahl sollte eines der Schlüsselwörter **minute**, **hour**, **day**, **week**, **month** oder **year** (oder deren Pluralformen) folgen. Das Schlüsselwort **next** kann als Synonym für **+ 1** verwendet werden.

Beispiele

Beachten Sie, daß die ersten beiden Kommandos gleichwertig sind:

```
at 1945 pm December 9
at 7:45pm Dec 9
at 3 am Saturday
at now + 5 hours
at noon next day
```

---

atq [*Optionen*]

Listet alle wartenden Jobs des Benutzers auf. Wenn der Superuser dieses Programm ausführt, werden die Jobs aller Benutzer aufgelistet. Das Programm entspricht dem Kommando **at -l**.

→

atq ←	*Optionen* -q *Warteschlange* Betrachtet nur die angegebene Warteschlange und ignoriert alle anderen Jobs. -v Zeigt die bereits ausgeführten, aber noch nicht aus der Warteschlange gelöschten Jobs an.
atrm	atrm *Job* [*Job* . . .] Löscht einen Job, der in der Warteschlange auf seine Ausführung wartet. Das Programm entspricht dem Kommando **at -d**.
banner	banner [*Option*] [*Zeichen*] Die *Zeichen* werden als Spruchband ausgegeben (132 Spalten breit, Schrift von oben nach unten). Wenn keine *Zeichen* auf der Kommandozeile angegeben sind, werden sie von der Standardeingabe gelesen. *Option* -w *Breite* Setzt die *Breite* zur Ausgabe des Spruchbandes auf den angegebenen Wert. Voreinstellung ist 132. Beachten Sie, daß die Ausgabe schmaler als der eingestellte Wert erscheint, wenn Sie nur Kleinbuchstaben verwenden.
bash	bash [*Optionen*] [*Datei* [*Argumente*]] sh [*Optionen*] [*Datei* [*Argumente*]] Standard Linux Shell, ein Kommandozeileninterpreter, in dem alle weiteren Kommandos eingegeben werden. Weitere Informationen finden Sie in Kapitel 3.
batch	batch [*Optionen*] [*time*] Führt die auf der Standardeingabe eingegebenen Kommandos aus. Wenn keine Zeit angegeben ist, wird mit der Ausführung begonnen, sobald es die Systemlast erlaubt (wenn die durchschnittliche Last unter 0,5 gefallen ist). Das Programm arbeitet ähnlich wie **at**, es besteht aber nicht auf dem Zeitpunkt, der für die Ausführung auf der Kommandozeile angegeben wurde. Sehen Sie auch unter **at**. *Optionen* -f *Datei* Liest die Jobs aus der *Datei* anstelle der Standardeingabe. -m Schickt dem Benutzer eine Mail, wenn der Job ausgeführt wurde, unabhängig davon, ob von dem Job eine Bildschirmausgabe erzeugt wurde oder nicht.

bc [*Optionen*] [*Dateien*]

bc ist eine Programmiersprache (und ein Compiler), deren Syntax der von C ähnlich ist, jedoch arithmetische Berechnungen mit unbegrenzter Genauigkeit durchführt. Der Sprachumfang von **bc** enthält Variablen, Schlüsselwörter und Symbole, die im folgenden kurz beschrieben werden. Am Ende werden einige Beispiele gegeben.

Das Programm führt interaktiv Berechnungen mit beliebiger Genauigkeit durch oder konvertiert Zahlen von einem Zahlensystem zu einem anderen. Die Eingabe kann aus *Dateien* oder von der Standardeingabe gelesen werden. Um das Programm zu verlassen, geben Sie **quit** oder das Symbol für **EOF** ein.

Optionen

-l	Ermöglicht die Benutzung von Funktionen aus der mathematischen Bibliothek.
-s	Alle Erweiterungen werden ignoriert und die Daten exakt nach dem POSIX-Standard verarbeitet.
-w	Wenn Erweiterungen vom POSIX-Standard für **bc** genutzt werden, erscheint eine Warnung.

Bezeichner

Ein Bezeichner ist eine Kette von einem oder mehr Zeichen. Er muß mit einem Kleinbuchstaben beginnen, darf aber auch Zahlen und Unterstriche enthalten. Großbuchstaben sind nicht erlaubt. Bezeichner werden als Namen für Variablen, Felder und Funktionen benutzt. Normalerweise können Variablen Zahlen mit beliebiger Genauigkeit aufnehmen. In einem Programm können Sie eine Variable, ein Feld und eine Funktion mit dem gleichen Buchstaben bezeichnen. Die folgenden Bezeichner führen zu keinem Konflikt:

x	Variable *x*.
x[*i*]	Element *i* des Feldes *x*. *i* darf aus dem Bereich 0 bis 2047 kommen und auch ein Ausdruck sein.
x(*y*,*z*)	Aufruf der Funktion *x* mit den Parametern *y* und *z*.

Schlüsselwörter für Ein- und Ausgabe

ibase, **obase**, **scale**, und **last** speichern einen Wert. Wenn sie allein auf einer Zeile eingegeben werden, wird der aktuelle Wert ausgegeben. Sie können den Wert auch durch eine Zuweisung verändern. Die Buchstaben A–F werden als Ziffern mit den Werten 10 bis 15 interpretiert.

ibase = *n*	Die Zahlen, die eingegeben werden (z.B. über die Tastatur), werden zur Basis *n* interpretiert (Voreinstellung ist 10).
obase = *n*	Zahlen werden zur Basis *n* dargestellt (Voreinstellung ist 10). Bemerkung: Nachdem die **ibase** von 10 verändert wurde, benutzen Sie die Ziffer A, um **ibase** oder **obase** auf das Dezimalsystem zu setzen.

→

scale = *n* Zeigt die Ergebnisse mit *n* Dezimalstellen an (Voreinstellung ist 0, die Resultate werden also auf Ganzzahlwerte gekürzt). **scale** wird normalerweise nur bei Berechnungen zur Basis 10 benutzt.

last Der Wert der zuletzt ausgegebenen Zahl.

Statements

Ein Semikolon oder ein Zeilenende trennt zwei Statements voneinander. Geschweifte Klammern werden benötigt, um mehrere Statements zusammenzufassen.

if (*Ausdruck*) {*Statements*} [**else** {*Statements*}]
Führt ein oder mehr *Statements* aus, wenn das Ergebnis vom *Ausdruck* wahr ist. Anderenfalls wird nichts gemacht, oder wenn ein *else*-Teil angegeben ist, werden die alternativen *Statements* bearbeitet. Zum Beispiel:

```
if(x==y) {i = i + 1} else {i = i - 1}
```

while (*Ausdruck*) {*Statements*}
Wiederholt ein oder mehr *Statements*, solange der *Ausdruck* wahr ist. Zum Beispiel:

```
while(i>0) {p = p*n; q = a/b; i = i-1}
```

for (*Ausdruck1*; *Bedingung*; *Ausdruck2*) {*Statements*}
Ähnlich wie **while**; um beispielsweise die ersten 10 Vielfachen von 5 auszugeben, tippen Sie:

```
for(i=1; i<=10; i++) i*5
```

GNU bc braucht nicht unbedingt drei Argumente für **for**. Wenn das erste oder das letzte Argument fehlt, werden diese Teile nicht ausgewertet. Wenn die Bedingung fehlt, wird sie immer als wahr bewertet.

break Unterbricht eine **while**- oder **for**-Schleife.

print *Liste*
GNU-Erweiterung; sie bietet eine zusätzliche Möglichkeit zur Ausgabe. Die *Liste* besteht aus einer durch Kommata separierten Reihe von Zeichenketten und Ausdrücken; **print** gibt die Werte in der Reihenfolge der Liste aus. Am Ende der Ausgabe wird kein Zeilenvorschub ausgegeben. Ausdrücke werden berechnet, ausgegeben und der speziellen Variablen *last* zugewiesen. Zeichenketten (die auch Sonderzeichen enthalten dürfen; zum Beispiel solche, die mit \ beginnen) werden einfach ausgegeben. Folgende Backslash-Symbole sind erlaubt:

a Alarm oder Glocke
b Backspace
f Seitenvorschub
n Zeilenvorschub
r Wagenrücklauf
q Anführungszeichen

t Tabulator

\ Backslash (\)

continue
GNU-Erweiterung. In einem **for**-Statement wird sofort mit dem nächsten Schleifendurchlauf begonnen.

halt GNU-Erweiterung. Veranlaßt, den **bc**-Programmablauf abzubrechen.

limits GNU-Erweiterung. Gibt die Wertebereiche der lokalen Version von **bc** aus.

Funktionen

define *j*(*args*) {
Beginn der Definition für die Funktion *j* mit den Argumenten *args*. Die Argumente werden durch Kommata getrennt. Die Statements folgen auf den nächsten Zeilen. Die Definition endet mit }.

auto *x*, *y*
Erzeugt die lokalen Variablen *x* und *y*, initialisiert mit dem Wert 0, die nur innerhalb der Funktion angesprochen werden können. Die Variablendeklaration muß am Anfang der Funktionsdefinition erfolgen.

return(*Ausdruck*)
Liefert den Wert vom *Ausdruck* zurück an das Programm. Wenn der *Ausdruck* weggelassen ist, wird 0 zurückgegeben. Sinnvoll nur in Funktionsdefinitionen.

sqrt(*Ausdruck*)
Berechnet die Quadratwurzel vom angegebenen *Ausdruck*.

length(*Ausdruck*)
Berechnet, wie viele signifikante Stellen das Ergebnis vom *Ausdruck* hat.

scale(*Ausdruck*)
Das gleiche wie **length**, es werden jedoch nur die Stellen vor dem Dezimalkomma gezählt.

read() GNU-Erweiterung. Liest eine Zahl von der Standardeingabe. Liefert den Wert der eingegebenen Zahl, interpretiert nach dem Wert von **ibase**.

Funktionen der mathematischen Bibliothek

Diese Funktionen stehen zur Verfügung, wenn das Programm **bc** mit der Option -l aufgerufen wurde. Die Bibliotheksfunktionen setzen **scale** auf 20.

s(*Winkel*)
Berechnet den Sinus vom *Winkel*, einer Konstante oder eines Ausdrucks in Bogenmaß.

c(*Winkel*)
Berechnet den Cosinus vom *Winkel*, einer Konstante oder eines Ausdrucks in Bogenmaß.

→

a(*n*) Berechnet den Arcustangens von *n* und liefert das Ergebnis in Bogenmaß.

e(*Ausdruck*)
Liefert die Potenz des Wertes vom *Ausdruck* zur Basis **e**.

l(*Ausdruck*)
Liefert den natürlichen Logarithmus vom *Ausdruck*.

j(*n*, *x*) Liefert den Wert der Bessel-Funktion der ganzzahligen Ordnung *n*.

Operatoren

Operatoren können arithmetisch, Vorzeichen, Zuweisungen oder Relationen sein:

Arithmetisch	+ - * / % ^
Vorzeichen	- ++ --
Zuweisung	=+ =- =* =/ =% =^ =
Relation	< <= > >= == !=

Andere Symbole

/* */ Schließen Kommentare ein.

( ) Kontrollieren die Reihenfolge, in der Unterausdrücke ausgewertet werden. Klammern können auch um eine Zuweisung gesetzt werden, um das Ergebnis des Ausdrucks ausgeben zu lassen.

{ } Geschweifte Klammern werden benutzt, um Statements zu einer Gruppe zusammenzufassen.

[ ] Eckige Klammern schließen den Index eines Feldes ein.

" *Text* "
In Anführungszeichen eingeschlossener *Text* wird ausgegeben.

Beispiele

Bitte beachten Sie, daß das Programm Werte, die Sie als Zahl oder als Ausdruck eingeben, berechnet und das Ergebnis ausgibt. Zuweisungen erzeugen jedoch keine Anzeige, wenn sie nicht in runden Klammern eingeschlossen sind.

ibase = 8	*Eingabe von Oktalzahlen*
20	*Berechne den (dezimalen) Wert dieser Oktalzahl*
16	*Auf dem Bildschirm erscheint die Dezimalzahl*
obase = 2	*Ausgabe wird umgestellt auf binär statt dezimal*
20	*Eingabe der Oktalzahl*
10000	*Auf dem Bildschirm erscheint jetzt die Binärzahl*
ibase = A	*Die Eingabebasis wird auf 10 zurückgesetzt*
scale = 3	*Die Genauigkeit wird auf drei Dezimalstellen begrenzt*
8/7	*Berechne die Division*
1.001001000	*Oops! Vergessen, die Ausgabebasis auf 10 zu setzen*
obase=10	*Die Eingabebasis ist schon 10, deshalb ist A hier nicht nötig*
8/7	
1.142	*Auf dem Bildschirm erscheint das Ergebnis (abgeschnitten)*

In den folgenden Zeilen wird die Benutzung von Funktionen gezeigt:

```
define p(r,n){   Funktion p benutzt zwei Argumente
  auto v         v ist eine lokale Variable
  v = r^n        n-te Potenz von r
  return(v)}     Der Wert wird zurückgegeben

scale=5
x=p(2.5,2)       x = 2.5 ^ 2
x                x wird ausgegeben
6.25
length(x)        Anzahl der Ziffern
3
scale(x)         Anzahl der Stellen nach dem Komma
2
```


biff [*Argumente*]

Der Benutzer wird bei ankommender Post benachrichtigt. **biff** arbeitet asynchron. Die Benachrichtigung funktioniert nur, wenn **sendmail** oder **smail** auf dem System läuft. Durch das Kommando **biff y** wird die Benachrichtigung aktiviert, **biff n** schaltet die Benachrichtigung aus. Ohne Argument wird der aktuelle Status von **biff** angezeigt.

cal [*–jy*] [[*Monat*] *Jahr*]

Gibt einen zwölfmonatigen Kalender (beginnend mit Januar) für ein bestimmtes *Jahr* oder einen Monatskalender für einen bestimmten *Monat* in dem *Jahr* aus. Die Monate werden mit den Monatszahlen 1 bis 12 angegeben, als Jahre sind die Zahlen 1 bis 9999 zugelassen. Ohne Argument wird der Kalender des aktuellen Monats angezeigt.

Optionen

- `-j` Gibt den julianischen Kalender aus (die Tage sind numeriert von 1 bis 365, beginnend am 1. Januar).
- `-y` Gibt den Kalender für das gesamte laufende Jahr aus.

Beispiele

```
cal 12 1995
cal 1994 > year_file
```


cat [*Optionen*] [*Dateien*]

Liest ein oder mehr *Dateien* und schreibt sie auf die Standardausgabe. Wenn keine *Dateien* angegeben sind oder wenn auf der Kommandozeile das Symbol anstelle eines Dateinamens übergeben wird, liest cat von der Standardeingabe. Sie können den Operator > benutzen, um mehrere Dateien aneinander zu hängen; mit >> hängen Sie Dateien an eine bereits existierende Datei an.

→

cat ←

Optionen

-e Gibt ein $-Zeichen am Ende jeder Zeile aus.

-n, --**number**
Gibt zu jeder Zeile die Zeilennummer am linken Rand aus, beginnend mit 1.

-s, --**squeeze-blank**
Unterdrückt die Ausgabe zusätzlicher Leerzeilen.

-t Ersetzt Tabulator- und Seitenvorschubzeichen durch die Zeichenfolge ^I bzw. ^L.

-u Wird ignoriert und nur zur Erhaltung der Abwärtskompatibilität mit UNIX angeboten.

-v, --**show-nonprinting**
Alle Steuerzeichen mit Ausnahme von LFD und TAB werden angezeigt.

-A, --**show-all**
Gleichbedeutend mit **-vET**: es werden alle Steuerzeichen, nichtdruckbaren Zeichen, Tabs und Zeilenenden dargestellt.

-E, --**show-ends**
Gibt ein $ am Ende jeder Zeile aus.

-T, --**show-tabs**
Gibt Tabulatoren als Zeichenkombination ^I aus.

Beispiele

`cat ch1`	*Gibt eine Datei aus*
`cat ch1 ch2 ch3 > all`	*Fügt mehrere Dateien zusammen*
`cat note5 >> notes`	*Hängt eine Datei an eine andere an*
`cat > temp1`	*Erzeugt eine neue Datei; Eingabe beenden mit EOF*
`cat > temp2 << STOP`	*Erzeugt eine neue Datei; Eingabe beenden mit* STOP

chattr

chattr [*Optionen*] *Modus Dateien*

Verändert die Dateiattribute von Dateien im Linux-Second-Extended-Dateisystem (EXT2). Das Programm arbeitet ähnlich wie **chmod** mit den Symbolen +, –, und =. Der *Modus* wird in der Form *Operator Attribut* angegeben.

Optionen

-R Verändert ganze Verzeichnisse mit ihrem Inhalt rekursiv.

-V Gibt Informationen über den Attributmodus nach der Veränderung aus.

-v *Version*
Setzt die Versionsnummer der Datei auf die angegebene Zahl.

Operatoren

\+ Fügt ein Attribut hinzu.

\- Löscht ein Attribut.

= Weist die angegebenen Attribute zu (und löscht alle übrigen).

Attribute

- a Append only, an die Datei können nur Daten angehängt werden.
- c Compressed, die Datei soll automatisch komprimiert werden (noch nicht vom Kernel unterstützt).
- d Die Datei wird bei einer Datensicherung mit dump ignoriert.
- i Immutable, die Datei kann nicht verändert werden.
- s Die Datei wird sicher gelöscht, das heißt, sie wird durch zufällige Daten überschrieben.
- u Undeletable, die Datei kann nicht gelöscht werden.
- S Der Lese-Cache und die auf der Platte gespeicherten Daten werden synchronisiert.

chfn [*Optionen*] [*Benutzername*]

NFS/NIS Programm — Ändert die in */etc/passwd* gespeicherte Information, die vom Finger-Dämon angezeigt wird. Ohne *Optionen* fragt **chfn** die Änderungen interaktiv ab. Um das gesamte Informationsfeld zu löschen, geben Sie das Schlüsselwort **none** ein.

Optionen

-f, --**full-name**
: Ändert den eingetragenen Namen des Benutzers.

-h, --**home-phone**
: Ändert den Eintrag für die private Telefonnummer des Benutzers.

-o, --**office**
: Ändert den Eintrag für die Büroadresse des Benutzers.

-p, --**office-phone**
: Ändert den Eintrag für die geschäftliche Telefonnummer des Benutzers

chgrp *Gruppe Dateien*

Ändert die Gruppenzugehörigkeit von einer oder mehr *Dateien* in die angegebene *Gruppe* um. Die *Gruppe* kann als numerische Group-ID oder als Gruppenname angegeben werden. Nur der Eigentümer einer Datei oder der Superuser kann die Gruppenzugehörigkeit ändern.

Optionen

-c, --**changes**
: Gibt Informationen über die durchgeführten Änderungen aus.

-f, --**silent**, --**quiet**
: Unterdrückt die Ausgabe von Fehlermeldungen, wenn Dateien nicht geändert werden können.

-R, --**recursive**
: Durchläuft rekursiv alle Unterverzeichnisse und führt die Änderungen an allen Dateien durch.

chmod

chmod [*Optionen*] *Modus Dateien*

Ändert den *Modus* für den Zugriff (Rechte) für eine oder mehr *Dateien*. Nur der Eigentümer einer Datei oder der Superuser darf den Zugriffsmodus ändern. Der *Modus* kann als numerischer oder als symbolischer Wert in der Form *Wer Operator Recht* angegeben werden. Der *Wer*-Teil ist optional (wenn er fehlt, wird **a** als Vorgabe verwendet). Sie können mehrere Rechte gleichzeitig angeben, dürfen aber zwischen dem *Wer*-Teil und den Rechten nur einen Operator verwenden. Es können mehrere symbolische Modi in einer durch Kommata separierten Liste angegeben werden.

Optionen

-c, --changes
: Gibt Informationen über die veränderten Dateien aus.

-f, --silent, --quiet
: Informiert den Benutzer nicht, wenn **chmod** eine Datei nicht ändern kann.

-v, --verbose
: Gibt Informationen über jede Datei aus, egal ob sie verändert wurde oder nicht.

-R, --recursive
: Geht rekursiv durch alle Unterverzeichnisse und führt die Modusänderung durch.

Wer

u	Benutzer
g	Gruppe
o	Andere (other)
a	Alle (Voreinstellung)

Operator

+	Fügt Recht hinzu
-	Entfernt ein Recht
=	Weist neue Rechte zu (und löscht die bisher geltenden Rechte)

Recht

r	Lesen (read)
w	Schreiben (write)
x	Ausführen (execute)
s	Setzt Benutzer- oder Gruppen-ID bei der Ausführung
t	Sticky bit; schützt Text (Datei) oder verhindert das Löschen einer Datei von Nicht-Eigentümern (Verzeichnis)
u	Die aktuellen Rechte des Benutzers
g	Die aktuellen Rechte der Gruppe
o	Die aktuellen Rechte der anderen
l	Mandatory Locking

Alternativ können die Rechte auch durch eine dreistellige Oktalzahl angegeben werden. Die erste Ziffer bezeichnet die Rechte des Eigentümers, die zweite die Gruppenrechte, und die dritte legt die Rechte der anderen Benutzer fest. Die Rechte werden errechnet, indem die folgenden Oktalwerte addiert werden:

4	Lesen
2	Schreiben
1	Ausführen

Bemerkung: Eine vierte Ziffer kann dieser Sequenz vorangehen. Diese Ziffer weist die folgenden Modi zu:

4	Setzt Benutzer-ID bei Ausführung.
2	Setzt Gruppen-ID bei Ausführung oder Mandatory Locking.
1	Setzt das Sticky Bit.

Beispiele

Erlaubt dem Eigentümer das Ausführen der *Datei*:

```
chmod u+x Datei
```

Jedes der folgenden Kommandos setzt die Rechte für die *Datei* so, daß der Eigentümer lesen, schreiben und ausführen (7), die Gruppe lesen und ausführen (5) und alle anderen nur ausführen (1) dürfen:

```
chmod 751 Datei
chmod u=rwx,g=rx,o=x Datei
```

Alle folgenden Kommandos weisen der *Datei* ausschließlich Leserecht für alle Benutzer zu:

```
chmod =r Datei
chmod 444 Datei
chmod a-wx,a+r Datei
```

Setzt die Benutzer-ID bei der Ausführung, erlaubt dem Eigentümer das Lesen, Schreiben und Ausführen und der Gruppe und den anderen das Lesen und Ausführen:

```
chmod 4755 Datei
```

chown *Neueigner Dateien*

Ändert den Eigentümer von einer oder mehr *Dateien* in den *Neueigner*. Der *Neueigner* kann entweder als numerische ID oder als Benutzername aus */etc/passwd* angegeben werden. **chown** erlaubt auch Benutzerbezeichnungen in der Form *Neueigner:Neugruppe*. Nur der aktuelle Eigentümer einer Datei oder der Superuser darf die Eigentumsverhältnisse ändern.

Optionen

-c, --changes
: Gibt Informationen über die veränderten Dateien aus.

→

chown ←

-f, --silent, --quiet
: Unterdrückt die Ausgabe von Fehlermeldungen über Dateien, die nicht verändert werden können.

-v, --verbose
: Gibt Informationen über alle Dateien aus, die **chown** zu ändern versucht, egal ob die Veränderung erfolgreich ist oder nicht.

-R, --recursive
: Geht rekursiv durch alle Unterverzeichnisse und führt die Veränderungen durch.

chsh

chsh [*Optionen*] [*Benutzername*]

Ändert die Loginshell interaktiv oder mit dem Kommandozeilenargument. Gibt eine Warnung aus, wenn die Shell nicht in */etc/shells* eingetragen ist.

Optionen

-l, --list-shells
: Gibt die erlaubten Shells aus, die in */etc/shells* aufgeführt sind.

-s, --shell
: Benutzt die auf der Kommandozeile angegebene Shell.

cksum

cksum [*Datei*]

Berechnet eine Prüfsumme (CRC) für alle *Dateien*, um auf Defekte zu prüfen. Liest von der Standardeingabe, wenn das Zeichen – oder gar keine Dateien angegeben sind. Gibt die Prüfsumme, die Dateigröße in Bytes und (wenn nicht von der Standardeingabe gelesen wurde) den Dateinamen aus.

clear

clear

Löscht den Terminalbildschirm.

cmp

cmp [*Optionen*] *Datei1 Datei2*

Vergleicht *Datei1* mit *Datei2*. Liest von der Standardeingabe, wenn *Datei1* als - bezeichnet ist oder fehlt. Vergleichen Sie auch **comm** und **diff**.

Optionen

-c, --print-chars
: Gibt die unterschiedlichen Bytes als Buchstaben aus.

-i *Anzahl*, --ignore-initial=*Anzahl*
: Überspringt die ersten *num* Bytes der Eingabe.

-l, --verbose
: Gibt Position und Inhalt aller differierenden Bytes aus.

-s, --quiet, --silent
: Arbeitet still; gibt nichts aus, aber gibt einen Rückgabewert an das aufrufende Progamm:

0 Die Dateien sind identisch
1 Die Dateien unterscheiden sich
2 Auf die Dateien kann nicht zugegriffen werden

Beispiel

Gibt eine Meldung aus, wenn zwei Dateien übereinstimmen (der Rückgabewert ist 0):

```
cmp -s old new && echo 'no changes'
```

---

colcrt [*Optionen*] [*Datei*]

Ein Filter zur Nachbereitung der Ausgabe von *tbl* (oder *nroff*) zur Behandlung von Escape-Sequenzen und Zeilenrückläufen, damit diese Ausgabe in brauchbarer Form auf einem Terminal dargestellt werden kann.

- Unterdrückt das Attribut „Unterstrichen".

-2 Stellt Halbzeilen dar, indem die Ausgabe mit doppeltem Zeilenabstand formatiert wird.

---

colrm [*Start* [*Stop*]]

Löscht die angegebenen Spalten aus einer Datei, wobei eine Spalte ein einzelnes Zeichen in einer Zeile ist. Das Programm liest von der Standardeingabe und schreibt in die Standardausgabe. Die Numerierung der Spalten beginnt mit 1. Der zu löschende Bereich beginnt in der durch *Start* und endet in der durch *Stop* angegebenen Spalte, jeweils einschließlich der begrenzenden Spalten. Ein Tabulator wird vollständig gelöscht, wenn der durch ihn erzeugte Raum in den zu löschenden Spaltenbereich ragt. Durch einen Backspace wird die Spaltenzahl um 1 erniedrigt.

---

column [*Optionen*] [*Datei*]

Formatiert die Daten (Zeilen) der Eingabe in Spalten, wobei die Elemente zuerst die Ausgabezeilen auffüllen. Es wird von der Standardeingabe gelesen, wenn keine *Datei* angegeben ist.

Optionen

-c *Anzahl*
Setzt die Gesamtbreite der Ausgabe auf die angegebene *Anzahl* Spalten.

-s *Zeichen*
Bestimmt das angegebene *Zeichen* als Trenner der Elemente in den Eingabedaten. Diese Option ist nur im Zusammenhang mit -t von Bedeutung.

-t Formatiert die Eingabedaten zu einer Tabelle. Die Elemente werden in einer Eingabezeile durch Leerzeichen getrennt, wenn nicht ein anderes Trennzeichen mit der Option -s festgelegt wurde. Zusammen mit -t sind die Optionen -c und -x wirkungslos.

→

column ←	-x Die Eingabedaten werden so angeordnet, daß zuerst die Spalten gefüllt werden.

comm

comm [*Optionen*] *Datei1 Datei2*

Vergleicht die beiden sortierten Dateien *Datei1* und *Datei2* zeilenweise. Das Ergebnis wird in drei Spalten ausgegeben: Zeilen, die nur in *Datei1* enthalten sind, Zeilen, die nur in *Datei2* enthalten sind, und Zeilen, die in beiden Dateien enthalten sind. **comm** hat Ähnlichkeit mit **diff**, da beide Programme zwei Dateien vergleichen. **comm** kann aber auch ähnlich wie **uniq** benutzt werden: **comm** sucht die doppelten bzw. die einzelnen Zeilen aus *zwei* sortierten Dateien, während **uniq** die doppelten bzw. einzelnen Zeilen in *derselben* sortierten Datei sucht.

Optionen

- Liest von der Standardeingabe.

-*Nummer*
: Unterdrückt die Ausgabe der Spalte *Nummer*. Es können mehrere Spalten gleichzeitig unterdrückt werden, indem die entsprechenden Ziffern ohne Trennzeichen angegeben werden.

Beispiele

Vergleicht zwei Kino-Hitlisten und gibt die Titel aus, die in beiden Listen erscheinen:

```
comm -12 siskel_top10 ebert_top10
```

cp

cp [*Optionen*] *Datei1 Datei2*
cp [*Optionen*] *Dateien Verzeichnis*

Kopiert *Datei1* nach *Datei2* oder kopiert ein oder mehr *Dateien* unter ihren aktuellen Namen in das *Verzeichnis*. Wenn das Ziel eine bereits existierende Datei ist, wird diese Datei überschrieben. Wenn das Ziel ein existierendes Verzeichnis ist, wird die Quelldatei in dieses Verzeichnis kopiert (das Verzeichnis wird *nicht* überschrieben).

Optionen

-a, --**archive**
: Erhält die Eigenschaften (Eigentümer, Rechte etc.) und die Verzeichnisstruktur der Quelldateien, soweit das möglich ist. Diese Option entspricht **-dpR**.

-b, --**backup**
: Erzeugt eine Sicherheitskopie von Dateien, die anderenfalls überschrieben würden.

-d, --**no-dereference**
: Ein symbolischer Link wird nicht aufgelöst, das heißt, es wird anstelle der Daten, auf die der Link verweist, der Link selbst unverändert kopiert.

-f, --force
Überschreibt existierende Zieldateien ohne Warnung.

-i, --interactive
Wartet vor dem Überschreiben einer Zieldatei auf eine Bestätigung.

-l, --link
Erzeugt Hardlinks anstelle von Kopien von normalen Dateien.

-p, --preserve
Erhält die Eigenschaften der Quelldatei, einschließlich Eigentümer, Gruppe, Zugriffsrechte und Zeitmarken.

-r
Kopiert normale Dateien und Unterverzeichnisse samt Inhalt rekursiv.

-s, --symbolic-link
Erzeugt symbolische Links anstelle von Kopien. Die Namen der Quelldateien müssen in diesem Fall mit dem gesamten, absoluten Pfadnamen angegeben werden.

-u, --update
Überschreibt eine existierende Zieldatei nur dann, wenn die Quelle die gleiche oder eine neuere Änderungszeit hat.

-v, --verbose
Gibt den Namen jeder Datei aus, bevor sie kopiert wird.

-x, --one-file-system
Ignoriert Unterverzeichnisse auf anderen Dateisystemen.

-P, --parents
Erhält die Verzeichnisstruktur der Quelldateien. Als Ziel muß ein existierendes Verzeichnis angegeben werden. Das Kommando **cp --parents jphekman/book/ch1 newdir** kopiert zum Beispiel die Datei *jphekman/book/ch1* nach *newdir/jphekman/book/ch1* und erzeugt automatisch alle dazwischenliegenden Verzeichnisse.

-R, --recursive
Kopiert Verzeichnisse mit Inhalt (auch Devices) rekursiv in das Zielverzeichnis.

-S, --suffix Endung
Legt die Endung fest, die beim Anlegen einer Sicherheitskopie an die Dateinamen angehängt wird. Diese Endung kann auch in der Umgebungsvariablen SIMPLE_BACKUP_SUFFIX angegeben werden. Die Voreinstellung ist ~.

-V, --version-control {numbered | existing | simple}
Stellt das Verhalten von cp beim Anlegen von Sicherheitskopien ein. Sie können dazu auch die Umgebungsvariable VERSION_CONTROL benutzen. Die Voreinstellung ist **existing**. Gültige Werte für die Versionskontrolle sind:

t, numbered Backups der gleichen Datei werden immer numeriert.

nil, existing Erzeugt numerierte Backups, wenn bereits ein numeriertes Backup existiert; anderenfalls wird ein einfaches Backup erzeugt (und ein

→

cp ←

	eventuell existierendes normales Backup überschrieben).
never, simple	Es werden immer einfache Backups gemacht.

cpio

cpio *Schalter* [*Optionen*]

Kopiert Dateiarchive zwischen Festplatte und Magnetband oder einem anderen Medium auf der lokalen Maschine hin oder her. Jeder der drei Schalter **-i**, **-o** oder **-p** eröffnet verschiedene zusätzliche Optionen:

-i [*Optionen*] [*Muster*]
Kopiert die Dateien vom Archiv in das Dateisystem hinein, deren Dateinamen mit dem angegebenen *Muster* übereinstimmen. Jedes Muster kann Sonderzeichen zur Dateinamenserweiterung enthalten, wie sie in der Bourne Shell verwendet werden. (Die Muster sollten in Anführungszeichen eingeschlossen oder durch Fluchtsymbole geschützt werden, damit sie von **cpio** und nicht von der Shell bewertet werden.) Wenn kein Muster angegeben ist, werden sämtliche Dateien aus dem Archiv in das Dateisystem hinein kopiert. Bei der Ausführung des Programms werden existierende Dateien nicht durch ältere Versionen aus dem Archiv überschrieben (es sei denn, die Option **-u** wird benutzt).

-o [*Optionen*]
Kopiert die Dateien aus dem Dateisystem auf das Archivmedium, deren Dateinamen auf der Standardeingabe benannt werden.

-p [*Optionen*] *Verzeichnis*
Kopiert Dateien an eine andere Stelle desselben Systems. Die Kopien werden mit relativen Pfadnamen im angegebenen *Verzeichnis* erzeugt.

Vergleich der gültigen Optionen

Die verfügbaren Optionen zu den drei Flags **-i**, **-o**, und **-p** werden jeweils in der ersten, zweiten und dritten Zeile unten dargestellt. (Das einleitende – wurde zugunsten der Übersichtlichkeit weggelassen.)

```
i:   bcdf mnrtsuv B SVCEHMR IF
o: 0a c          vABL VC HM O F
p: 0a  d lm     uv   L V    R
```

Optionen

-0, --null
Die Dateinamen der Liste werden durch Nullbytes anstelle eines Zeilenendes voneinander getrennt.

-a, --reset-access-time
Setzt das Datum des letzten Zugriffs auf die Dateien auf den Wert vor der Archivierung zurück.

-A, --append
Hängt Dateien an ein existierendes Archiv an, das sich auf einer Festplattenpartition oder einer Diskette befinden muß. Der Name des Archivs wird mit **-O** oder **-F** angegeben.

-b, --swap
Die Bytes in Datenwörtern und Halbwörtern werden getauscht.

-B Die Blockgröße der Ausgabe wird auf 5120 Bytes erhöht (Voreinstellung sind 512 Bytes).

-c Liest oder schreibt Dateiinformationen im Archivkopf als ASCII-Zeichen; sinnvoll, wenn Quell- und Zielmaschine von unterschiedlichem Typ sind.

-C *n*, **--io-size=***n*
Wie **-B**, die Blockgröße kann aber durch einen beliebigen ganzzahligen Wert *n* frei gewählt werden.

-d, --make-directories
Erzeugt bei Bedarf neue Verzeichnisse.

-E *Datei*, **--pattern-file=***Datei*
Extrahiert Dateien, deren Namen der angegebenen *Datei* aufgelistet bzw. durch Muster beschrieben sind.

-f, --nonmatching
Kehrt die Bedeutung des Musters um; es werden nur Dateien extrahiert, deren Name *nicht* auf das *Muster* paßt.

-F, --file=*Datei*
Benutzt die angegebenen *Datei* als Archiv, nicht die Standardein- oder Standardausgabe. Die *Datei* kann sich auf einem anderen Rechner befinden, wenn der Name in der Form *Benutzer@Hostname:Datei* angegeben wird.

--force-local
Unterdrückt den Versuch, auf eine Archivdatei auf einer entfernten Maschine zuzugreifen (deren Name durch eine der Optionen **-F**, **-I**, oder **-O** angegeben wurde), auch wenn der Name einen Doppelpunkt oder ein @ enthält.

-H *Typ*, **--format=***Typ*
Benutzt den angegebenen *Typ* für das Archivformat. Die Voreinstellung bei der Erzeugung neuer Archive ist **bin**. Gültige Formate sind (Angaben in Großbuchstaben sind ebenfalls erlaubt):

bin	Binärformat
odc	Altes (POSIX.1-konformes) portables Format
newc	Neues (SVR4-konformes) portables Format
crc	Neues (SVR4-konformes) portables Format mit zusätzlicher Prüfsumme
tar	Tar-Format
ustar	POSIX.1-tar-Format (es werden auch GNU-Tar-Archive erkannt)
hpbin	Das Binärformat von HP-UX (veraltet)
hpodc	Das portable Archivformat von HP-UX

-I *Datei*
Liest das Archiv aus der angegebenen *Datei*. Archivdateien auf entfernten Rechnern sind möglich (siehe Option **-F**).

-k Ohne Funktion (für Abwärtskompatibilität).

→

-l, --link
Erzeugt Links anstelle von Kopien.

-L, --dereference
Folgt den symbolischen Links.

-m, --preserve-modification-time
Bewahrt die gespeicherte Änderungszeit der Dateien.

-M *Meldung*, **--message=***Meldung*
Gibt die *Meldung* aus, wenn das Speichermedium zur Archivierung gewechselt werden muß. Durch die Variable %d kann eine fortlaufende Nummer in die Meldung eingefügt werden. Die Option **-M** kann nur in Verbindung mit **-I** oder **-O** benutzt werden.

-n, --numeric-uid-gid
Bei einem ausführlichen Listing des Archivinhalts werden Benutzer- und Gruppen-ID in numerischer Form angezeigt.

--no-preserve-owner
Beim Extrahieren von Dateien aus dem Archiv oder beim Kopieren wird die archivierte Benutzer- und Gruppen-ID nicht auf die extrahierten Daten übertragen. (Voreinstellung für Benutzer ohne root-Privilegien.)

-O *Datei*
Lenkt die Ausgabe in die angegebene *Datei* um. Es können auch Archive auf entfernten Rechnern erzeugt werden (siehe Option -F).

-r Die Dateien können interaktiv umbenannt werden.

-R [*Benutzer*][:.][*Gruppe*], **--owner** [*Benutzer*][:.][*Gruppe*]
Weist den Dateien beim Kopieren den angegebenen Eigentümer und die Gruppe zu (nur für Benutzer mit root-Privilegien möglich).

-s, --swap-bytes
Die Bytes eines Daten-Halbwortes werden getauscht.

-S, --swap-half-words
Die Halbwörter eines Datenwortes werden getauscht.

-t, --list
Gibt ein Inhaltsverzeichnis eines Dateiarchivs aus (es werden keine Dateien kopiert). Zusammen mit der Option **-v** wird eine Ausgabe in der Art von **ls -l** erzeugt.

-u, --unconditional
Unbedingte Kopie; alte Dateien aus dem Archiv können neuere Dateien im Dateisystem überschreiben.

-v Gibt bei der Arbeit eine Liste der bearbeiteten Dateien aus.

-V, --dot
Gibt bei der Arbeit einen Punkt für jede bearbeitete Datei aus. (Das zeigt den Fortschritt von **cpio** an, ohne dabei den Bildschirm durcheinander zu bringen.)

Beispiele

Erzeugt mit Hilfe von **find** eine Liste alter Dateien und benutzt diese Liste als Eingabe für **cpio**:

```
find . -name "*.old" -print | cpio -ocBv\
    > /dev/rst8
```

Restauriert alle Dateien, deren Namen das Wort **save** enthalten, von einem Bandarchiv (Unterverzeichnisse werden bei Bedarf erzeugt):

```
cpio -icdv "save" < /dev/rst8
```

Kopiert einen Verzeichnisbaum:

```
find . -depth -print | cpio -padm /mydir
```

csh [*Optionen*] [*Datei* [*Argumente*]]

Die C-Shell, ein Kommandozeileninterpreter, in dem alle weiteren Kommandos eingegeben werden. Mehr Information finden Sie in Kapitel 4.

csplit [*Optionen*] *Datei Argumente*

Zerteilt die angegebene *Datei* in Abschnitte und legt jeden Abschnitt einer eigenen Datei unter einem Namen der Form **xx00** bis **xx***n* (*n* < 100) ab. Die Trennstellen werden durch Muster festgelegt, die durch *Argumente* auf der Kommandozeile angegeben werden. Siehe auch **split**.

Optionen

-
: Liest die Daten von der Standardeingabe.

-b, --suffix-format=*Endung*
: Hängt die *Endung* an die Ausgabedateinamen an. Die *Endung* muß festlegen, wie eine Binärzahl in ein lesbares Format umgewandelt wird, und genau eines der folgenden Symbole enthalten: %d, %i, %u, %o, %x oder %X. Das Symbol wird interpretiert wie eine Formatanweisung von printf:

  %d (Vorzeichenbehaftete) Dezimalzahl

  %i Entspricht %d

  %u (Vorzeichenlose) Dezimalzahl

  %o Oktalzahl

  %x Hexadezimalzahl

  %X Entspricht %x. Die Verwendung dieser Option macht den Schalter -n wirkungslos.

-f *Prefix*, **--prefix=***Prefix*
: Die Namen der erzeugten Dateien beginnen mit dem angegebenen *Prefix* anstelle des voreingestellten **xx**.

→

-k, --keep-files

Behält die erzeugten Dateien, auch wenn bei der Ausführung des Programms ein Fehler auftritt (der normalerweise zur Löschung aller bereits erzeugten Dateien führen würde). Diese Option ist sinnvoll, wenn Sie eine beliebig hohe Anzahl von Wiederholungen angeben und verhindern wollen, daß ein `out of range`-Fehler zur Löschung der neuen Dateien führt.

-n, --digits=*Anzahl*

Erzeugt Ausgabedateinamen mit der angegebenen *Anzahl* Stellen in der numerischen Endung. Voreinstellung ist 2.

-s, -q, --silent, --quiet

Unterdrückt die Ausgabe der Zeichenzahl der einzelnen Abschnitte.

-z, --elide-empty-files

Leere Ausgabedateien werden nicht erzeugt, bei der Numerierung weiterer Ausgabedateien werden sie jedoch mitgezählt.

Argumente

Als Argumente können alle folgenden Ausdrücke einzeln und in Kombinationen verwendet werden. Ausdrücke, die Leerzeichen oder andere Sonderzeichen enthalten, müssen in einfache Anführungszeichen eingeschlossen werden.

/*Ausdruck*/[*Offset*]

Erzeugt eine Datei mit den Daten von aktuellen Zeile bis zu einer Zeile, die den regulären *Ausdruck* enthält. Ein *Offset* kann in der Form +*n* oder -*n* angegeben werden, wobei *n* die Anzahl der Zeilen vor oder nach dem *Ausdruck* ist.

%*Ausdruck*%[*Offset*]

Das Gleiche wie /*Ausdruck*/, mit dem Unterschied, daß keine Datei mit den Daten der Zeilen bis zur Zeile mit dem *Ausdruck* erzeugt wird.

Anzahl

Erzeugt eine Datei mit der angegebenen *Anzahl* Zeilen, beginnend in der aktuellen Zeile. Wenn dem Argument ein Wiederholungsbefehl folgt (eine Zahl innerhalb von {}), wird jeweils die angegebene Anzahl Zeilen in einer anderen Datei gespeichert.

{*n*}
Wiederholt das letzte Argument *n*-mal. Der Wiederholungsbefehl kann jedem der oben genannten Argumente folgen. Dateien werden an den Stellen mit dem *Ausdruck* oder in Blöcken zu *Anzahl* Zeilen getrennt. Wenn ein * anstelle von *n* angegeben ist, wird das Argument so oft wiederholt, bis die Eingabedaten vollständig verarbeitet sind.

Beispiele

Erzeugt bis zu 20 Kapiteldateien aus einer Datei **novel**:

```
csplit -k -f chap. novel '%CHAPTER%' '{20}'
```

Erzeugt aus einer Datei **address_list** bis zu 100 Adreßdateien (**xx00** bis **xx99**) mit einer Länge von jeweils vier Zeilen.

```
csplit -k address_list 4 {99}
```

csplit

cut *Optionen* [*Dateien*]

cut

Schneidet bestimmte Spalten oder Felder aus den angegebenen *Dateien* aus. Bei den folgenden Optionen wird als *Liste* eine Folge von ganzzahligen Werten akzeptiert. Einzelne Werte werden durch Komma voneinander getrennt, Bereiche können durch einen Bindestrich angegeben werden (z.B. **1-10,15,20** oder **50-**). Sehen Sie auch **paste**, **join** und **newform**.

Optionen

-b, --**bytes** *Liste*
: Legt eine *Liste* von Positionen fest: Nur Bytes in diesen Positionen werden ausgegeben.

-c, --**characters** *Liste*
: Es werden die Zeichen in den durch die *Liste* angegebenen Positionen ausgegeben.

-d, --**delimiter** *c*
: Wird zusammen mit -f benutzt, um den Feldtrenner *c* festzulegen (Voreinstellung ist tab); Sonderzeichen (z.B. Leerzeichen) müssen in Anführungszeichen angegeben werden.

-f, --**fields** *Liste*
: Gibt die in der *Liste* festgelegten Felder aus.

-s, --**only-delimited**
: Zusammen mit -f wird die Ausgabe von Zeilen unterdrückt, in denen das Trennzeichen nicht vorkommt.

Beispiele

Extrahiert Benutzer- und Real-Namen aus der Datei **/etc/passwd**:

```
cut -d: -f1,5 /etc/passwd
```

Findet heraus, wer gerade eingeloggt ist, und listet nur die Login-Namen auf.

```
who | cut -d" " -f1
```

Schneidet die Buchstaben aus der vierten Spalte der *Datei* aus und fügt sie vor die erste Spalte der gleichen Datei ein:

```
cut -c4 Datei | paste - Datei
```

date [*Optionen*] [+*Format*] [Datum]

date

Schreibt oder setzt Systemzeit und -datum. Es kann ein Ausgabe*format* angegeben werden. Das Ausgabe*format* besteht aus Zeichenfolgen und Felddeskriptoren, die im folgenden beschrieben werden (logisch sortiert).

→

date ←

Optionen

+*Format*
: Das Datum soll benutzerdefiniert ausgegeben werden. Zum Beispiel:

```
% date +"%A E %j %n%k %p"
Tuesday 248
15 PM
```

Voreingestellt ist %a %b %e %T %Z %Y – z.B. Tue Sep 5 14:59:37 MET 1995.

-d *Datum*, --**date** *Datum*
: Gibt das *Datum* anstelle des aktuellen Datums aus. *Datum* kann auch das Format *d* **days** oder *m* **months** *d* **days** haben, um ein zukünftiges Datum auszugeben. Mit **ago** kann ein vergangenes Datum ausgegeben werden. Auch hier kann ein Ausgabenformat angegeben werden, siehe hierzu den Abschnitt „Format".

-s *Datum*, --**set** *Datum*
: Setzt Datum und Zeit. Darf nur der Superuser.

-u, --**universal**
: Veranlaßt die Interpretation von Datum und Zeit nicht als lokale Zeit, sondern als Greenwich Mean Time (GMT).

Format

%%	Ein % ausgeben.
-	Datumsfelder nicht auffüllen (voreingestellt: von links mit Nullen füllen).
_	Datumsfelder mit Leerzeichen auffüllen (statt mit Nullen).
%a	Wochentag in abgekürzter Schreibweise.
%b	Monatsname in abgekürzter Schreibweise.
%c	Datum und Zeit in landesüblicher Darstellung.
%d	Tag im Monat (01 bis 31).
%h	Wie %b (Monat, abgekürzt).
%j	Tag im Jahr (Julianisches Datum: Tag von 001 bis 366).
%k	Stunde (24-Stunden-Format, ohne führende Nullen: 0 bis 23).
%l	Stunde (12-Stunden-Format, ohne führende Nullen: 1 bis 12).
%m	Monatsnummer (01 bis 12).
%n	Ein Zeilenvorschub.
%p	Zeichenkette zur Anzeige von Vor- oder Nachmittag (voreingestellt: AM und PM).
%r	Abkürzung für %I:%M:%S %p (12-Stunden-Format).
%s	Sekunden seit dem 1970-01-01 00:00:00 (Unix-Zeit; Epoche).
%t	Ein Tabulator.
%w	Tag der Woche (Sonntag = 0).
%x	Datum in landesüblicher Darstellung.
%y	Jahreszahl mit zwei Stellen (00 bis 99).
%A	Wochentag in voller Länge.
%B	Monatsname in voller Länge.

%D Abkürzung für %m/%d/%y (6+2-stelliges Datum).
%H Stunde (24-Stunden-Format: 00 bis 23).
%I Stunde (12-Stunden-Format: 01 bis 12).
%M Minuten (00 bis 59).
%S Sekunden (00 bis 59).
%T Abkürzung für %H:%M:%S (6+2-stellige Zeit).
%U Wochennummer im Jahr (00 bis 53) mit Wochenbeginn am Sonntag.
%W Wochennummer im Jahr (00 bis 53) mit Wochenbeginn am Montag.
%X Zeit in landesüblicher Darstellung.
%Y Jahreszahl mit vier Stellen (z.B. 1996).
%Z Kurzbezeichnung der Zeitzone (z.B. MET) oder leer, falls Zone nicht feststellbar.

Gültige Angaben zum Setzen von Datum und Zeit

Das Standardformat zum Setzen besteht aus Monat, Tag, Stunde, Minute, Jahr im Format *mmddHHMMyy*. Sekunden können als *.SS* angehängt werden. Das Jahr kann zwei- oder vierstellig angegeben werden, die Angabe der Stunden erfolgt im 24-Stunden-Format. Weiterhin können die Felder von rechts angefangen weggelassen werden. Der Monat darf mit seinem englischen Namen angegeben werden, die Angabe einer Zeitzone ist optional.

Beispiele

Datum auf 1. Juli (**0701**), 18.42 Uhr (**1842**), 1995 (**95**) setzen:

```
date 0701184295
date Jul01184295
```

Das Kommando:

```
date +"Salut,%t Datum ist %D %n%t Zeit ist %T"
```

erzeugt eine Ausgabe der Form:

```
Salut,       Datum ist 05/09/93
             Zeit ist 17:53:39
```

df [*Optionen*] [*Verzeichnis*]

Zeigt den freien Plattenplatz (disk free) aller aufgesetzten Dateisysteme. Die Ausgabe erfolgt nur für aufgesetzte Dateisysteme. Ist *Verzeichnis* angegeben, wird nur der Platz des Dateisystems angegeben, in dem sich *Verzeichnis* befindet. Wenn *Verzeichnis* der Name einer Gerätedatei (z.B. */dev/hd**) oder Pfad eines Aufsetzpunktes ist (z.B. */usr*), dann wird der freie Platz des Gerätes oder des an diesem Aufsetzpunkt aufgesetzten Dateisystems ausgegeben. Angaben erfolgen in Kilobyte, es sei denn, die Variable POSIXLY_CORRECT ist gesetzt: Dann erfolgen die Angaben in freien 512-Byte-Sektoren.

→

df
←

Optionen

-a, --**all** Zeigt auch Dateisysteme an, die leer sind (0 Bytes Kapazität).

-i, --**inodes**
Zeigt die Auslastung der Inodes an: freie, benutzte und Prozent-belegte Inodes.

-k, --**kilobytes**
Erzwingt die Angabe des freien Platzes in Kilobyte, ignoriert die Variable POSIXLY_CORRECT.

-t, --**type=***Typ*
Zeigt nur Dateisysteme des Typs *Typ* an.

-x, --**exclude-type=***Typ*
Nimmt Dateisysteme des Typs *Typ* von der Anzeige aus.

-P, --**portability**
Erzwingt Ausgabe im POSIX-Format: genau eine Zeile pro Dateisystem ausgeben.

-T, --**print-type**
Gibt zu jedem Dateisystem den zugehörigen Typ an, z.B. zur Benutzung für **-t** oder **-x**.

---

diff

diff [*Optionen*] [*DirOptionen*] *Datei1 Datei2*

Vergleicht zwei Textdateien. **diff** gibt diejenigen Zeilen aus, in denen sich *Datei1* und *Datei2* unterscheiden. Zu den ausgegebenen Zeilen wird der Zusammenhang angegeben, zu dem sie gehören. Standardmäßig werden Zeilen aus *Datei1* durch ein < markiert und solche aus *Datei2* mit >. Den Zeilen stehen Befehle des Editors **ed** voran (**a**, **c**, or **d**), mit denen man *Datei1* in *Datei2* umwandeln könnte. Wird für eine der Dateien - angegeben, wird die Standardeingabe gelesen. Wenn eine der Dateien ein Verzeichnis ist, sucht **diff** die zum anderen Dateinamen passende Datei im angegebenen Verzeichnis. (Z.B. ist **diff mein_verz txtfile** das gleiche wie **diff mein_verz/txtfile txtfile**.) Falls beide Dateien Verzeichnisse sind, bearbeitet **diff** diejenigen Dateien beider Verzeichnisse, die gleiche Namen haben (z.B. *altverz/text1* und *neu_verz/text1*); weiterhin führt **diff** gemeinsame Unterverzeichnisse und die Dateien auf, die sich jeweils nur in einem der Verzeichnisse befinden. Siehe auch **cmp**.

Optionen

-b, --**ignore-space-change**
Behandelt wiederholte Leerzeichen wie eins und ignoriert Leerzeichen an Zeilenenden.

-e, --**ed**
Liefert als Ausgabe eine Befehlsfolge für den Editor **ed** (**a**, **c**, **d**), mit der *Datei2* in *Datei1* überführt werden kann.

-H Erhöht die Ausgabegeschwindigkeit bei langen Dateien, indem gezielt nach verstreuten kleinen Veränderungen gesucht wird. Weiträumige Änderungen werden nicht als solche ausgegeben.

-i, --**ignore-case**
Behandelt Klein- und Großbuchstaben gleich.

-w, --ignore-all-space
Zieht Leerzeichen überhaupt nicht zum Vergleich heran.

-B, --ignore-blank-lines
Ignoriert Leerzeilen in den Dateien.

-I *regexp*, **--ignore-matching-lines=***regexp*
Ignoriert Zeilen, die dem regulären Ausdruck *regexp* entsprechen.

-a, --text
Behandelt alle Dateien als Textdateien. So kann auch mit **diff** festgestellt werden, ob Binärdateien denselben Inhalt haben.

-c Context-**diff**: anderes Ausgabeformat, welches die Zeilen in einem Textzusammenhang von ± 3 Zeilen ausgibt.

-u Unified-**diff**: kompaktere Form von **-c**, in welcher alte und neue Zeilen in nur einem Block ausgegeben werden.

-q, --brief
Es wird nur ausgegeben, *ob* sich die Dateien unterscheiden.

-n, --rcs
Erzeugt Ausgaben im Format, welches **RCS** erwartet (RCS diff).

-y, --side-by-side
Erzeugt die Ausgabe in zwei Spalten.

-w *n* Gibt die maximale Ausgabebreite für das Zwei-Spalten-Format (-y) an, voreingestellt ist 130.

--left-column
Gibt im Zwei-Spalten-Format (-y) bei gleichen Zeilen nur die linke Spalte aus.

--suppress-common-lines
Unterdrückt im Zwei-Spalten-Format (-y) die Ausgabe gleicher Zeilen.

-l, --paginate
Läßt die Ausgabe von **pr** in Seiten unterteilen.

-t, --expand-tabs
Läßt die Tabulatoren in der Ausgabe zu entsprechend vielen Leerzeichen umwandeln.

-T, --initial-tab
Fügt einen Tabulator am Zeilenanfang an, um bereits vorhandene Tabulatoren korrekt anzuordnen.

-r, --recursive
Läßt Unterverzeichnisse rekursiv weitervergleichen.

-N, --new-file
Behandelt fehlende Dateien als leer.

-s, --report-identical-files
Läßt angeben, wenn sich Dateien *nicht* unterscheiden.

-x *regexp*, **--exclude=***regexp*
Unterläßt den Vergleich von Dateien in einem Verzeichnis, welches dem regulären Ausdruck *regexp* entspricht.

→

diff ←

-X *Dateiname*, --**exclude-from=***Dateiname*
Unterläßt den Vergleich von Dateien in Verzeichnissen, welche regulären Ausdrücken aus der Datei *Dateiname* entsprechen.

-S *Dateiname*
Beginnt beim Vergleich von Verzeichnissen mit der Datei *Dateiname*.

-d, --**minimal**
Versucht intensiv, die Anzahl der ausgegebenen Änderungen gering zu halten.

---

diff3

diff3 [*Optionen*] *DateiNeu1 DateiAlt2 DateiNeu3*

Vergleicht drei Dateien und gibt die Unterschiede wie folgt aus:

====	Alle Dateien sind unterschiedlich.
====1	*DateiAlt1* unterscheidet sich.
====2	*DateiNeu2* unterscheidet sich.
====3	*DateiAlt3* unterscheidet sich.

diff3 hat die Aufgabe, Veränderungen in zwei Dateien zu erkennen, die von einer gemeinsamen Grundversion stammen, z.B. wenn zwei Personen unabhängig voneinander Veränderungen an einer Datei vorgenommen haben. **diff3** kann die Veränderungen zwischen der Grundversion und einer der neuen Dateien erkennen und erzeugt als Ausgabe die Änderungen, die an der anderen neuen Datei vorgenommen wurden, um alle Veränderungen zu berücksichtigen.

Konkret wird *DateiAlt2* mit *DateiNeu3* verglichen, und diese Unterschiede werden auf *DateiNeu1* bezogen.

Es wird zwischen folgenden Begriffen unterschieden: *Unvereinte Änderungen* (Textstellen, wo sich die neuen Dateien unterscheiden und mindestens eine auch von der Grundversion verschieden ist), *vereinte Änderungen* (Änderungen gegenüber der Grundversion, die den neuen Daten gemeinsam sind) sowie *überlappende Änderungen* (Textstellen, die in allen Dateien verschieden sind).

Optionen

-3
Läßt eine **ed**-Befehlsfolge ausgeben, die alle unvereinten, nicht-überlappenden Änderungen zwischen *DateiNeu1* und *DateiNeu3* in *DateiNeu1* aufnimmt.

-a, --**text**
Behandelt alle Dateien als Textdateien.

-A, --**show-all**
Läßt eine **ed**-Befehlsfolge aller Änderungen ausgeben; überlappende Änderungen werden eingerahmt ausgegeben.

-e, --**ed**
Läßt eine **ed**-Befehlsfolge ausgeben, die alle unvereinten Änderungen zwischen *DateiNeu1* und *DateiNeu3* in *DateiNeu1* aufnimmt.

-E, --**show-overlap**
Läßt eine **ed**-Befehlsfolge aller unvereinten Änderungen ausgeben; überlappende Änderungen werden eingerahmt ausgegeben.

-x, --**overlap-only**
Läßt eine **ed**-Befehlsfolge aller überlappenden Änderungen ausgeben, die in *DateiNeu1* aufgenommen werden kann.

-X
Wie -x, überlappende Änderungen werden eingerahmt ausgegeben.

-m, --**merge**
Läßt das Ergebnis aller Änderungen als Datei ausgeben.

-L *benenn*, --**label**=*benenn*
Benutzt in der Ausgabe *benenn* als Dateinamen.

-i
Nimmt zusätzlich die Befehle **w** (save) und **q** (quit) in die **ed**-Befehlsfolge auf, um **ed** mit Sichern zu verlassen.

-T, --**initial-tab**
Fügt einen Tabulator am Zeilenanfang ein, um bereits vorhandene Tabulatoren korrekt anzuordnen.

du [*Optionen*] [*Verzeichnis* ...]

Zeigt den belegten Plattenplatz (disk usage) für das *Verzeichnis* und dessen Unterverzeichnisse in Kilobyte-Blöcken an. Ohne Argumente erfolgt die Angabe für das Arbeitsverzeichnis.

Optionen

-a, --**all**
Zeigt die Belegung für alle Dateien (nicht nur Verzeichnisse) an.

-b, --**bytes**
Veranlaßt die Verwendung von Bytes als Ausgabeeinheit.

-c, --**total**
Gibt zum Schluß zusätzlich die Gesamtsumme aller Argumente aus.

-k, --**kilobytes**
Läßt die Ausgabe in Kilobytes erfolgen (Voreinstellung).

-l, --**count-links**
Zählt Dateien, auf die durch Links mehrfach verwiesen wird, entsprechend häufig mit.

-r
Gibt die Meldung `cannot open` aus, wenn auf eine Datei nicht zugegriffen werden kann.

-s, --**summarize**
Gibt nur die Belegung der *Verzeichnis*se an, unterläßt die Angaben für die Unterverzeichnisse.

-x, --**one-file-system**
Ignoriert Verzeichnisse, die in anderen Dateisystemen liegen.

-D, --**dereference-args**
Löst symbolische Links auf, jedoch nur für die in der Kommandozeile angegebenen *Verzeichnis*se.

→

du ←

-L, --dereference
: Löst symbolische Links auf, gibt also die Belegung der Dateien an, auf die die symbolischen Links verweisen.

-S, --separate-dirs
: Zählt die Belegung der Unterverzeichnisse nicht zu der Belegung ihrer Eltern hinzu.

---

dumpkeys

dumpkeys [*Optionen*]

Gibt Angaben zu den Tastaturbelegungstabellen aus. Siehe auch die Beschreibung zu *keytables* in den Manpages.

Optionen

-c*Zeichensatz*, --**charset**=*Zeichensatz*
: Wählt den Zeichensatz aus, mit dem die Belegung ausgegeben werden soll; zulässig sind: **iso-8859-1** (Voreinstellung), **iso-8859-2**, **iso-8859-3**, **iso-8859-4** und **iso-8859-8**.

--**compose-only**
: Gibt nur die Kombinationen für die Compose-Taste aus.

-f, --**full-table**
: Lange Ausgabe: zu jeder Taste wird ihre Belegung unter allen Modifiern ausgegeben.

--**funcs-only**
: Gibt nur die Definitionen der Funktionstasten aus.

-i, --**short-info**
: Kurzes Info: Angaben zu erlaubten keycode keywords in den Tastaturtabellen, der Anzahl der zulässigen Aktionen für eine Taste, eine Liste der erlaubten Bereiche der action codes (die rechten Seiten der Tastendefinitionen) sowie die Anzahl der vom Kernel unterstützten Funktionstasten.

--**keys-only**
: Gibt nur die einfachen Tastenbelegungen aus.

-l, --**long-info**
: Langes Info: Zusätzlich zum kurzen Info wird eine Liste der action symbols und ihrer Zahlenwerte ausgegeben.

-n, --**numeric**
: Gibt die Werte der key actions hexadezimal aus, keine Umsetzung auf symbolische Namen.

---

echo

echo [-n] [*Argumente*]

Gibt seine *Argumente* aus. **echo** ist in viele Shells fest eingebaut, hier wird das Kommando unter **/bin/echo** beschrieben. Einige Zeichenkombinationen werden wie folgt umgesetzt:

Optionen

\a Alert (bell) – Signal

\b Backspace – Rücktaste

\c Unterdrückt den abschließenden Zeilenvorschub

\f Form feed – Seitenvorschub

\n Newline – Zeilenvorschub

\r Carriage return – Wagenrücklauf

\t Horizontal tab – waagerechter Tabulator

\v Vertical tab – senkrechter Tabulator

\\ Literal backslash – das Zeichen \

nnn Druckt das ASCII-Zeichen mit dem Oktalwert *nnn*.

-n Unterdrückt den abschließenden Zeilenvorschub

Beispiele

```
echo "Ein Test für den Drucker" | lp
echo "TITEL\nTITEL" > file ; cat doc1 doc2 >> file
echo "Achtung: Signalton \07"
```


egrep [*Optionen*] [*RegAus*] [*Dateien*]

Durchsucht eine oder mehrere *Dateien* nach Zeilen, die dem regulären Ausdruck *RegAus* entsprechen. Einige reguläre Ausdrücke werden von **egrep** nicht unterstützt: \(, \), *n*, \<, \>, \{ und \}, dafür aber zusätzlich die Ausdrücke +, ?, | und ( ). Zur Erinnerung: Diese Zeichen haben spezielle Bedeutungen in Shells und sollten entsprechend geschützt werden. Eine Beschreibung regulärer Ausdrücke findet sich in Kapitel 5. Der Rückgabewert ist 0, falls Zeilen ausgegeben wurden, 1, falls nicht, sowie 2 bei Fehlern. Siehe auch bei **grep** und **fgrep**. **egrep** ist typischerweise der schnellste Befehl der **grep**-Familie.

Optionen

-*num* Gibt *num* Zeilen vor und nach einer Fundstelle aus, es wird jedoch keine Zeile öfter als einmal ausgegeben.

-b Gibt zu jeder Zeile ihre Blocknummer aus.

-c (count) Gibt nur die **Anzahl** der gefundenen Zeilen aus.

-e *RegAus* (expression.) Sucht nach dem Suchmuster *RegAus*. Diese Option ist sinnvoll für Suchmuster, die mit einem - anfangen.

-f *AusdrDatei*
(file) Entnimmt den Suchausdruck der Datei *AusdrDatei*.

-i (ignore case) Behandelt Groß- und Kleinbuchstaben gleich.

-l gibt nur **Namen** der Dateien aus, in denen Zeilen gefunden wurden.

-h Gibt ausschließlich die gefundenen Zeilen aus und nicht die Namen der Dateien.

→

egrep ←

-n	Gibt die Zeilennummern zu den gefundenen Zeilen aus.
-q	Unterdrückt die normale Ausgabe (quiet).
-s	Unterläßt Fehlermeldungen über Dateien, die nicht geöffnet werden können (silent).
-w	Wendet das Suchmuster nur auf ganze Wörter an. Wörter werden durch Zeichen begrenzt, die weder Buchstabe, Zahl noch der Unterstrich sind.
-x	Wendet das Suchmuster nur auf ganze Zeilen an.
-v	Gibt alle Zeilen aus, die *nicht* dem Suchmuster entsprechen.
-A *num*	(after) Gibt zusätzlich *num* Zeilen vor einer Fundstelle aus.
-B *num*	(before) Gibt zusätzlich *num* Zeilen nach einer Fundstelle aus.
-C	(context) Gibt zusätzlich zwei Zeilen vor und nach jeder Fundstelle aus.
-L	Gibt die Dateien aus, in denen *keine* Zeilen gefunden wurden.

Beispiele

Nach allen Vorkommen von *Victor* oder *Victoria* in *Datei* suchen:

```
egrep 'Victor(ia)*' Datei
egrep '(Victor|Victoria)' Datei
```

Zeichenketten der Art *old.doc1* oder *new.doc2* in *Dateien* suchen und mit Zeilennummern ausgeben:

```
egrep -n '(old|new)\.doc?' Dateien
```

emacs

emacs [*Optionen*] [*Dateien*]

Ein Editor zum normalen Bearbeiten von Text sowie für alle Dinge, die sonst noch bei der Arbeit anfallen könnten. Eine genaue Beschreibung findet sich in Kapitel 6.

env

env [*option*] [*variable*=*Wert* ...] [*Kommando*]

Gibt alle Umgebungsvariablen aus. Wenn die *Variable* angegeben ist, wird sie vorübergehend auf *Wert* gesetzt, und die so geänderten Umgebungsvariablen werden ausgegeben. Wenn ein *Kommando* angegeben ist, wird es unter der so geänderten Umgebung ausgeführt.

Optionen

-, -i, --ignore-environment
Ignoriert die bisher bestehenden Umgebungsvariablen.

-u *name*, **--unset** *name*
Nimmt die Setzung der angegebenen Umgebungs*variable* zurück.

ex [*Optionen*] *Dateien*

ex

Ein interaktiver kommandogestützter Editor. Eine genauere Beschreibung befindet sich in Kapitel 8.

expand [*Optionen*] *Dateien*

expand

Wandelt die Tabulatoren in den angegebenen Dateien (– steht für die Standardeingabe) in die entsprechende Anzahl von Leerzeichen um und gibt das Ergebnis auf der Standardausgabe aus.

Optionen

-, -t, --tabs *Tabulatoren*
: *Tabulatoren* ist eine Zahlenliste (durch Kommata getrennt), welche die Position der Tabulatorwerte angibt. Falls genau eine Zahl angegeben wird, werden Tabulatoren in diesem gleichmäßigen Abstand gesetzt. Voreingestellt sind Tabulatoren in einem Abstand von acht Leerzeichen. Bei **-t** und **--tabs** kann die Zahlenliste auch durch Leerzeichen getrennt sein.

-i, --initial
: Ersetzt Tabulatoren nur an Zeilenanfängen.

expr *arg1 operator arg2* [*operator arg3* ...]

expr

expr wertet seine Argumente als Ausdruck aus, um das Resultat auszugeben. Die Bestandteile des Ausdrucks müssen getrennt werden, im Normalfall durch Leerzeichen. Argumente sind normalerweise ganze Zahlen, explizit angegeben oder das Ergebnis der Auswertung einer Shell-Variablen. Es gibt drei Operator-Typen: arithmetische, relationale und logische. Der Rückgabewert von **expr** ist **0** (Ausdruck ist weder leer noch null), **1** (Ausdruck ist leer oder null) oder **2** (ungültiger Ausdruck).

Arithmetische Operatoren

Diese Operatoren ermöglichen die Durchführung der Grundrechenarten

+ Addiert *arg2* zu *arg1*.

- Subtrahiert *arg2* von *arg1*.

* Multipliziert die Argumente.

/ Dividiert *arg1* durch *arg2*.

% Nimmt den Divisionsrest von *arg1* dividiert durch *arg2*.

Addition und Subtraktion werden zuletzt durchgeführt, falls sie nicht in runde Klammern eingeschlossen sind. Bei den Operatoren *, (, und) sollte man ihre spezielle Bedeutung für Shells beachten und sie entsprechend schützen.

Relationale Operatoren

Diese Operatoren ermöglichen Vergleiche zwischen ihren Argumenten. Dies können auch Zeichenketten sein, hier erfolgt der Vergleich dann länderspezifisch. Ist ein Vergleichsausdruck wahr, wird 1 geliefert, sonst 0. Achtung: > und < haben spezielle Bedeutung in Shells.

→

=, ==	Testet auf Gleichheit.
!=	Testet auf Verschiedenheit.
>	Testet, ob *arg1* größer als *arg2* ist.
>=	Testet, ob *arg1* größer als *arg2* oder gleich oder gleich ist.
<	Testet, ob *arg1* kleiner als *arg2* ist.
<=	Testet, ob *arg1* kleiner als *arg2* oder gleich ist.

Logische Operatoren

Auch diese Operatoren ermöglichen Vergleiche. Das Resultat kann *arg1* (oder ein Teil davon), *arg2* oder 0 sein. Achtung: **|** und **&** haben spezielle Bedeutung in Shells.

|
: Logisches ODER: Liefert *arg1*, wenn *arg1* weder null noch leer ist, sonst *arg2*.

&
: Logisches UND: Liefert *arg1*, wenn *arg1* und *arg2* weder null noch leer sind, sonst 0.

:
: Wie **grep**: *arg2* dient als Suchmuster, das in *arg1* gesucht wird. *arg2* muß ein regulärer Ausdruck sein. Falls ein Teil des Musters aus *arg2* in \(\) eingeschlossen ist, wird der Teil von *arg1* geliefert, der dem Ausdruck entspricht. Anderenfalls wird die Anzahl der Zeichen geliefert, die dem Ausdruck entsprechen. Das Suchmuster beginnt immer am Wortanfang, deshalb muß der Suchausdruck mit .* anfangen, um anderen Wortteilen zu entsprechen.

match *Zeichenkette* **regexp**
: Wie *Zeichenkette : regexp.*

substr *Zeichenkette Start Länge*
: Liefert die Teilkette von *Zeichenkette* ab Position *Start* mit der Länge *Länge.* Liefert Null, wenn *Start* oder *Länge* negativ oder keine Zahl ist.

index *Zeichenkette Zeichenliste*
: Liefert die erste Position in *Zeichenliste,* an der ein Zeichen aus *Zeichenliste* vorkommt, oder 0, falls kein Zeichen aus *Zeichenliste* in *Zeichenkette* vorkommt.

length *Zeichenkette*
: Liefert die Länge von *Zeichenkette.*

Beispiele

Punktrechnung geht vor Strichrechnung, das Ergebnis ist 10:

```
expr 5 + 10 / 2
```

Klammern gehen vor, das Ergebnis ist 7 (7,5 ohne Nachkommastellen):

```
expr \( 5 + 10 \) / 2
```

Zu der Variablen **i** 1 hinzuzählen. Typischerweise werden so Variablen in Shell-Skripten hochgezählt.

```
i=`expr $i + 1`
```

Gibt 1 (für wahr) aus, falls die Variable **a** „hello" ist:

```
expr $a = hello
```

Gibt 1 (für wahr) aus, falls die Variable **b** plus 5 größergleich 10 ist

```
expr $b + 5 \>= 10
```

In den folgenden Beispielen ist die Variable **p** gleich „version.100". So wird die Anzahl der Zeichen in **p** ausgegeben: **p**:

```
expr $p : '.*'
```
Ergebnis ist 11

Alle Zeichen suchen und ausgeben:

```
expr $p : '\(.*\)'
```
Ergebnis ist „version.100"

Die Anzahl der aufeinanderfolgenden Kleinbuchstaben am Anfang von **p** ausgeben:

```
expr $p : '[a-z]*'
```
Ergebnis ist 7

Die aufeinanderfolgenden Kleinbuchstaben am Anfang von **p** ausgeben:

```
expr $p : '\([a-z]*\)'
```
Ergebnis ist „version"

$x auf fünf Zeichen kürzen, falls es fünf Zeichen oder mehr hat; sonst einfach **$x** ausgeben (das logische ODER nimmt das zweite Argument, falls das erste 0 ist).

```
expr $x : '\(.....\)' \| $x
```

In einem Shell-Skript einen Dateinamen auf die fünf ersten Buchstaben kürzen:

```
mv $x `expr $x : '\(.....\)' \| $x`
```

(Um das Überschreiben von vorhandenen Dateien zu vermeiden, ggf. **mv -i** benutzen.)

false

Liefert ohne weitere Aktion einen Rückgabewert wie ein fehlerhaft gelaufenes Programm (ungleich null). Nützlich für Shell-Skripten. Siehe auch **true**.

fdformat [*Optionen*] *Gerätedatei*

Eine Diskette im Diskettenlaufwerk formatieren. Die zugehörigen Gerätedateien sind gewöhnlich **/dev/fd0** und **/dev/fd1**.

Option

-n Unterläßt die abschließende Erfolgskontrolle.

fgrep

fgrep [*Optionen*] [*Suchmuster*] [*Dateien*]

Durchsucht eine oder mehrere *Dateien* nach Zeilen, welche die Zeichenkette *Suchmuster* enthalten. Der Rückgabewert ist **0**, falls Zeilen ausgegeben wurden, **1**, falls nicht, sowie **2** bei Fehlern. Siehe auch bei **egrep** und **grep**.

Optionen

-b
: Gibt zu jeder Zeile ihre Blocknummer aus.

-c
: Gibt nur die **Anzahl** der gefundenen Zeilen aus (count).

-e *Suchmuster*
: (Expression.) Sucht nach *Suchmuster*. Diese Option ist sinnvoll für Suchmuster, die mit einem - anfangen.

-f *MustDatei*
: Entnimmt eine Liste von Suchmustern der Datei *MustDatei* (file).

-h
: Gibt ausschließlich die gefundenen Zeilen aus und nicht die Namen der Dateien.

-i
: Behandelt Groß- und Kleinbuchstaben gleich (ignore case).

-l
: Gibt nur **Namen** der Dateien aus, in denen Zeilen gefunden wurden.

-n
: Gibt die Zeilennummern zu den gefundenen Zeilen aus.

-*num*
: Gibt *num* Zeilen vor und nach einer Fundstelle aus.

-v
: Gibt alle Zeilen aus, die *nicht* dem Suchmuster entsprechen.

-w
: Wendet das Suchmuster nur auf ganze Wörter an. Wörter werden durch Zeichen begrenzt, die weder Buchstabe, Zahl noch der Unterstrich sind.

-x
: Wendet das Suchmuster nur auf ganze Zeilen an.

-A *num*
: Gibt zusätzlich *num* Zeilen vor einer Fundstelle aus (after).

-B *num*
: Gibt zusätzlich *num* Zeilen nach einer Fundstelle aus (before).

-C
: Gibt zusätzlich zwei Zeilen vor und nach jeder Fundstelle aus (context).

-L
: Gibt die Dateien aus, in denen *keine* Zeilen gefunden wurden.

Beispiele

Alle Zeilen aus *Datei* ausgeben, die keine Leerzeichen enthalten:

```
fgrep -v ' ' Datei
```

Alle Zeilen aus *Datei* ausgeben, die Wörter aus der Datei **RechtschreibLexikon** enthalten:

```
fgrep -f RechtschreibLexikon Datei
```

file

file [*Optionen*] *Datei* ...

file stellt den Typ einer *Datei* anhand des Anfangs ihrer Daten fest. Hierfür nutzt es die Informationen aus der Datei */etc/magic* (oder */usr/lib/magic*).

Optionen

- -c Überprüft das Format der Magic-Datei. (Das Argument *Datei* ... ist hier nicht zulässig.)
- -f *Liste*
  Wendet **file** auf die Namen an, die in der Datei *Liste* stehen.
- -m *AltMagic*
  Sucht Informationen in *AltMagic* statt in */etc/magic*.
- -z Versucht, weitere Informationen zu komprimierten Dateien zu erhalten.
- -L Folgt symbolischen Links. Standardmäßig werden sie nicht verfolgt.

file erkennt viele Typen. Die Ausgabe erfolgt zeilenweise als Dateiname, gefolgt vom erkannten Dateityp, zum Beispiel:

```
ascii text
c program text
c-shell commands
data
empty
iAPX 386 executable
directory
[nt]roff, tbl, or eqn input text
shell commands
symbolic link to ../usr/etc/arp
```

Beispiel

Alle Dateien suchen, die troff/nroff-Text zu enthalten scheinen:

```
file * | grep roff
```

find [*Pfadnamen*] [*Bedingungen*]

find ist nützlich, um nach verschiedenen Kriterien eine Gruppe von Dateien zusammenzustellen. Hierfür wird der Verzeichnisbaum bei jedem *Pfadnamen* beginnend abwärts durchsucht, um Dateien zu finden, die den angegebenen *Bedingungen* entsprechen. Als Pfadname ist das Arbeitsverzeichnis voreingestellt. Als besonders nützliche Bedingungen seien vorab erwähnt: **-print** (voreingestellt, falls keine andere Bedingung angegeben), **-name** und **-type** (einfache Suche), **-exec** und **-size** (erweiterte Suche) und **-mtime** und **-user** (zu administrativen Zwecken).

Bedingungen können durch runde Klammern \(\) gruppiert, durch ! negiert (\! in der C-Shell!), durch **-o** verodert sowie wiederholt werden. Letzteres ist sinnvoll, um weitere Einschränkungen anzugeben, z.B. bei **-name**, **-type** und **-perm**.

Bedingungen und Aktionen

- **-atime** +*n* | -*n* | *n*
  Findet Dateien, die mindestens vor *n* (+*n*) Tagen, innerhalb der letzten *n* (-*n*) Tage oder vor genau *n* Tagen gelesen wurden.

→

Hinweis: **find** ändert die Access-Time aller Verzeichnisse, die als *Pfadnamen* angegeben wurden.

-ctime +*n* | -*n* | *n*

Findet Dateien, deren Inode mindestens vor *n* (+*n*) Tagen, innerhalb der letzten *n* (-*n*) Tage oder vor genau *n* Tagen geändert wurde. Ein Inode wird durch jede Aktion geändert, die den Eintrag einer Datei im Verzeichnis ändert, wie z.B. **chmod**.

-depth

Findet beim Abstieg durch den Verzeichnisbaum die Dateien eines Verzeichnisses *vor* dem Verzeichnis selbst. Dadurch werden Dateien in tieferliegenden Verzeichnissen vor Dateien in höherliegenden Verzeichnissen gefunden. Dies ist nützlich, wenn Dateien in Verzeichnissen liegen, die nicht beschrieben werden dürfen, wenn z.B. **find** mit **cpio** zusammen benutzt wird.

-exec *Kommando* { } \;

Führt das UNIX-*Kommando* mit jeder von **find** gefundenen Datei aus (wird das bezeichnete *Kommando* mit dieser Datei erfolgreich ausgeführt, gibt es 0 als Rückkehrstatus zurück). Wenn das *Kommando* ausgeführt wird, ersetzt das Argument { } den aktuellen Dateinamen. Die ganze Sequenz wird mit einem geschützten Semikolon (\;) abgeschlossen.

-follow

Folgt symbolischen Links und merkt sich die aufgesuchten Verzeichnisse (nicht mit **-type l** verwenden).

-group *gname*

Sucht Dateien, die der mit *gname* bezeichneten Gruppe gehören. *gname* kann entweder der Gruppenname oder die Gruppen-ID-Nummer sein.

-inum *n*

Sucht die Dateien mit der Inode-Nummer *n*.

-links *n*

Sucht Dateien, die *n* Links haben.

-mount, -xdev

Sucht nach Dateien, die sich im gleichen Dateisystem wie *Pfadname* befinden.

-mtime +*n* | -*n* | *n*

Findet Dateien, die vor mehr als *n* (+*n*), weniger als *n* (-*n*) oder exakt *n* Tagen zuletzt modifiziert wurden. Eine Modifizierung ist eine Änderung im Datenkörper der Datei.

-name *Muster*

Sucht nach Dateien, deren Bezeichnung mit dem *Muster* übereinstimmen. Dateinamen mit Meta-Zeichen können im Muster verwendet werden, müssen aber vor der Shell geschützt werden.

-newer *Datei*

Sucht Dateien, die später als *Datei* modifiziert wurden, ähnlich wie **-mtime**. **-follow** berücksichtigt diese Bedingung nur, wenn sie nach **-follow** auf der Kommandozeile erscheint.

-ok *Kommando* { } \;

Entspricht **-exec**, verlangt aber vom Benutzer vor der Ausführung des *Befehls* eine Bestätigung in Form von **y**.

-perm *nnn*

Sucht die Dateien, deren Berechtigungen (z.B. **rwx**) der oktalen Ziffer *nnn* exakt entsprechen (z.B. 664 entspricht **-rw-rw-r--**). Durch Voranstellen eines Minuszeichens werden alle Dateien gefunden, die mindestens die angegebenen Berechtigungs-Bits gesetzt haben (z.B. **-perm -600** entspricht **-rw*********, wobei * jeder Modus sein kann).

-print Gibt die gefundenen Dateien und Verzeichnisse mit ihren vollen Pfadnamen aus. Liefert den Wahrheitswert **true**.

-size *n*[c]

Sucht Dateien, die *n* Blöcke enthalten oder, wenn **c** angegeben ist, *n* Zeichen lang sind.

-type *c*

Sucht Dateien vom Typ *c*. *c* kann **b** (blockorientierte Gerätedatei), **c** (zeichenorientierte Gerätedatei), **d** (Verzeichnis), **p** (FIFO oder Named Pipe), **l** (symbolische Links) oder **f** (einfache Datei) sein.

-user *Benutzer*

Sucht Dateien, die dem mit Namen oder ID bezeichneten *Benutzer* gehören.

-daystart

Veranlaßt die Berechnung von Zeiten vom Beginn des aktuellen Tages, nicht von der aktuellen Zeit an.

-maxdepth *num*

Beschränkt die Suchetiefe auf die angegebene Anzahl von Verzeichnisebenen.

-mindepth *num*

Läßt die Suche erst ab der angegebenen Tiefe von Verzeichnisebenen beginnen.

-noleaf

Erzwingt die Bearbeitung aller Verzeichniseinträge. Normalerweise kann davon ausgegangen werden, daß jedes Verzeichnis wenigstens zwei harte Links enthält (das Verzeichnis selbst und das Elternverzeichnis), die nicht durchsucht werden müssen. **-noleaf** unterdrückt diese Annahme, was in solchen Fällen sinnvoll ist, in denen **find** auf Nicht-UNIX-Dateisystemen arbeitet.

-amin +*n* | -*n* | *n*

Sucht Dateien, auf die vor mehr als *n* (+*n*), weniger als *n* (-*n*) oder genau *n* Minuten zugegriffen wurde.

-anewer *Datei*

Sucht Dateien, auf die nach der letzten Änderung zugegriffen wurde. **-follow** berücksichtigt diese Bedingung nur, wenn sie nach **-follow** auf der Kommandozeile erscheint.

→

-cmin +*n* | -*n* | *n*
Sucht Dateien, die vor mehr als *n* (+*n*), weniger als *n* (-*n*) oder genau *n* Minuten verändert wurden.

-cnewer *Datei*
Sucht Dateien, deren Inode nach der letzten Änderung des Dateikörpers verändert wurde. **-follow** berücksichtigt diese Bedingung nur, wenn sie nach **-follow** auf der Kommandozeile erscheint.

-empty
Sucht alle leeren Dateien. Wird sowohl auf reguläre Dateien als auch auf Verzeichnisse angewendet.

-false Liefert immer den Wahrheitswert FALSCH für jede gefundene Datei.

-fstype *Typ*
Sucht nur in Dateisystemen des angegebenen *Typs*. Erlaubte Dateisystemtypen sind unter anderem **minix**, **msdos**, **ext**, **ext2** und **proc**.

-gid *num*
Sucht Dateien mit der numerischen Gruppen-ID *num*.

-ilname *Muster*
Entspricht **-lname** ohne Unterscheidung von Groß- und Kleinschreibung.

-iname *Muster*
Entspricht **-name** ohne Unterscheidung von Groß- und Kleinschreibung.

-ipath *Muster*
Entspricht **-path** ohne Unterscheidung von Groß- und Kleinschreibung.

-iregex *Muster*
Entspricht **-regex** ohne Unterscheidung von Groß- und Kleinschreibung.

-lname *Muster*
Sucht nach symbolischen Links, die auf Dateien zeigen, deren Namen mit dem angegebenen *Muster* übereinstimmen. Das *Muster* kann Jokerzeichen in Shellsyntax enthalten. / oder . werden nicht besonders behandelt. Groß- und Kleinschreibung wird ignoriert.

-mmin +*n* | -*n* | *n*
Sucht Dateien, die vor mehr als *n* (+*n*), weniger als *n* (-*n*) oder genau *n* Minuten verändert wurden.

-nouser
Sucht Dateien, deren Eigentümer keine gültige ID auf dem System haben.

-nogroup
Sucht Dateien, deren Gruppe keine gültige ID auf dem System hat.

-path *Muster*

Sucht Dateien, deren Namen auf das angegebene *Muster* passen. Es werden absolute Pfadnamen erwartet, / und . haben keine besondere Bedeutung.

Beispiele

Listet alle Dateien (und Unterverzeichnisse) des Heimatverzeichnisses auf:

```
find $HOME -print
```

Listet alle Dateien mit dem Namen *chapter1* im Verzeichnis */work* auf:

```
find /work -name chapter1 -print
```

Listet alle Dateien von *ann*, deren Name mit *memo* beginnt, auf:

```
find /work /usr -name 'memo*' -user ann -print
```

Durchsucht das Dateisystem (vom Wurzelverzeichnis an) nach Verzeichnissen für Manualpages:

```
find / -type d -name 'man*' -print
```

Durchsucht das aktuelle Verzeichnis nach Dateinamen, die nicht mit einem Großbuchstaben beginnen, und schickt sie zum Drucker:

```
find . \! -name '[A-Z]*' -exec lp {} \;
```

Sucht und komprimiert Dateien, deren Namen (noch) nicht mit .gz enden:

```
gzip `find . \! -name '*.gz' -print`
```

Löscht alle leeren Dateien aus dem System (interaktiver Modus):

```
find / -size 0 -ok rm {} \;
```

Durchsucht das System nach Dateien, die innerhalb der letzten zwei Tage modifiziert wurden (als gute Kandidaten für ein Backup):

```
find / -mtime -2 -print
```

Sucht mittels **grep** rekursiv in einem Verzeichnisbaum nach einem Muster:

```
find /book -print | xargs grep '[Nn]utshell'
```

finger [*Optionen*] *BenutzerName*

Informationen über einen oder mehrere *BenutzerNamen* ausgeben, unter anderem die Inhalte der Dateien **.plan** und **.project** im Home-Verzeichnis jedes *BenutzerNamens*. Der *BenutzerName* kann entweder die Benutzerkennung (genaue Angabe) oder der Vor- oder Nachname sein (Angabe über alle passenden Namen). Innerhalb eines Netzwerkes sind auch *Benutzernamen* der Form *user@host* und *@host* zulässig.

→

finger ←

Optionen

- -l Langes Ausgabeformat (Voreinstellung): wie -s, zusätzlich Angaben zu Home-Verzeichnis, Privattelefon, Login-Shell, gelesene Post, den Dateien **.plan**, **.project** und **.forward**.
- -m Unterläßt die Ausgabe der Realnamen.
- -p Unterläßt die Ausgabe der Dateien **.plan** und **.project**.
- -s Kurzes Ausgabeformat: Benutzerkennung, Realname, Terminalname, Status zu **write**, Zeit der letzten Aktivität, Bürolage und -telefonnummer.

fmt

fmt [*Optionen*] [*Datei* ...]

(format) Setzt den Text der Dateien im Blocksatz der angegebenen Breite. Standardgemäß werden Leerzeilen, Position von Leerzeichen und der Einzug am Zeilenanfang erhalten. **fmt** versucht, Zeilenumbrüche an Satzenden durchzuführen sowie sie nach dem ersten oder vor dem letzten Wort eines Satzes zu vermeiden.

Optionen

-c, **--crown-margin**
: Weist an, die Einrückung der ersten beiden Zeilen eines Absatzes zu erhalten und diejenige der zweiten Zeile für den Rest des Absatzes zu verwenden.

-p *prefix*, **--prefix**=*prefix*
: Formatiert nur Zeilen, die mit *prefix* anfangen.

-s, **--split-only**
: Entfernt keine Zeilenumbrüche der Eingabe.

-t, **--tagged-paragraph**
: Wie -c, falls die Einrückungen der ersten beiden Zeilen eines Absatzes verschieden sind. Sonst wird die erste Zeile als eigener Absatz angesehen.

-u, **--uniform-spacing**
: Erzwingt genau ein Leerzeichen zwischen Wörtern und zwischen zwei Sätzen.

-width *Breite*, -w *Breite*, **--width**=*Breite*
: Setzt die Ausgabebreite auf *Breite*. Voreingestellt ist 75.

fold

fold [*option*] [*Datei* ...]

Fügt Zeilenumbrüche derart in die Eingabedateien ein, daß die entstehenden Zeilen nicht länger als die eingestellte Breite (voreingestellt: 80) sind. **fold** fügt die Umbrüche exakt bei der eingestellten Breite ein, selbst wenn es mitten im Wort ist. Liest von der Standardeingabe, falls - als Dateiname angegeben wird.

Optionen

-b, **--bytes**
: Zählt einzelne Zeichen und nicht Ausgabespalten, d.h., Tabula-

toren, Rückschritte und Wagenrückläufe zählen als genau ein Zeichen.

-s, --spaces
: Führt den Umbruch – soweit möglich – nur an Leerzeichen durch.

-w, --width *Breite*
: Setzt *Breite* als maximale Zeilenlänge.

formail [*Optionen*]

Filtert die Standardeingabe in ein gültiges Mailbox-Format. Wenn kein Absender erkannt wird, setzt das Programm statt dessen *foo@bar* ein. Ohne weitere Kommandozeilenargumente werden fehlerhafte **From**:-Zeilen durch ein > am Zeilenanfang entwertet.

Optionen

+*skip* Unterdrückt die Trennung der angegebenen Anzahl Nachrichten am Begin der Eingabe.

-*total* Beendet die Trennung der Eingabedaten nach der angegebenen Anzahl Nachrichten.

-a *headerfield*
: Hängt das angegebene *headerfield* an den Nachrichtenkopf an, wenn dieses Feld nicht bereits existiert. Wenn der Inhalt des Feldes Message-ID oder Resent-Message-ID ohne weitere Angaben ist, wird automatisch eine eindeutige Message-ID erzeugt und eingetragen.

-b Unterdrückt die Entwertung von fehlerhaften **From**-Zeilen.

-c Wenn Felder im Nachrichtenkopf mehr als eine Zeile lang sind, werden sie zu einer Zeile zusammengefaßt.

-d Erlaubt die Verarbeitung von Eingabedaten, die nicht im strikten Mailbox-Format vorliegen.

-e Erlaubt die Aneinanderreihung von Nachrichten ohne die sonst übliche Leerzeile zur Trennung.

-f Unterdrückt die Veränderung von Zeilen, die dem Mailbox-Format widersprechen. Beispielsweise wird keine **From**-Zeile am Anfang jeder Nachricht eingefügt.

-i *headerfield*
: Hängt das *headerfield* an den Nachrichtenkopf an, egal ob ein solches Feld bereits existiert oder nicht. Ein existierendes Feld wird in **Old**-*headerfield* umbenannt, wenn es nicht leer ist.

-k Zur Benutzung zusammen mit der Option **-r**. Der Nachrichtenkörper bleibt ebenso wie die durch **-r** bestimmten Felder erhalten.

-m *minfields*
: Erwartet die angegebene Mindestzeilenzahl, bevor der Anfang einer neuen Nachricht erkannt wird. Voreinstellung ist 2.

→

formail
←

-n Erlaubt mehrere parallele **formails** zur Verarbeitung einzelner Nachrichten.

-p *prefix*
Bestimmt das angegebene *prefix* anstelle des voreingestellten > zur Entwertung fehlerhafter Kopfzeilen.

-q Unterdrückt die Ausgabe von Meldungen über Schreibfehler, doppelte Nachrichten und unzutreffende **Content-Length**-Felder. Das ist die Voreinstellung, benutzen Sie -q-, um die Meldungen zur Anzeige zu bringen.

-r Alle existierenden Felder bis auf **X-Loop** werden verworfen und ein neuer Auto-Reply-Header erzeugt. Einzelne Felder können mit der Option **-i** geschützt werden.

-s Diese Option kennzeichnet das Ende der Optionsliste. Alle darauffolgenden Kommandozeilenargumente werden als Kommando interpretiert, dem die einzelnen Nachrichten auf der Standardeingabe übergeben werden.

-t Veranlaßt das Akzeptieren der Return-Adresse (normalerweise bevorzugt **formail** maschinengenerierte Adressen).

-u *headerfield*
Das erste auftretende Feld des angegebenen Typs wird unverändert weitergegeben, alle weiteren werden gelöscht.

-x *headerfield*
Gibt den Inhalt des Feldes auf einer einzelnen Zeile aus.

-z Wenn nötig, werden Leerzeichen zwischen Feldname und Feldinhalt eingefügt und leere Felder entfernt.

-A *headerfield*
Hängt das *headerfield* an den Nachrichtenkopf an, egal ob es bereits existiert oder nicht.

-B Die Eingabedaten werden im BABYL-**rmail**-Format erwartet.

-D *maxlen idcache*
Die Message-IDs werden in einer Datei *idcache* der maximalen Größe *maxlen* gespeichert, und durch Vergleich wird ermittelt, ob die aktuelle Nachricht bereits verarbeitet wurde. Wenn das Programm die Nachrichten nicht teilt, wird der Wahrheitswert TRUE zurückgeliefert, wenn eine doppelte Nachricht gefunden wurde. Wenn das Programm die Nachrichten trennt, werden doppelte Nachrichten nicht ausgegeben. Zusammen mit der Option **-r** wird die Mailadresse des Absenders anstelle der Message-ID verwendet.

-I *headerfield*
Hängt das *headerfield* an den Nachrichtenkopf an, egal ob ein gleichnamiges Feld bereits existiert oder nicht. Ein existierendes Feld wird gelöscht.

-R *oldfield newfield*
Wandelt alle Feldnamen *oldfield* um in *newfield*.

-U *headerfield*
Löscht alle Felder mit dem angegebenen Namen bis auf das letzte.

-Y Veranlaßt die Formatierung im Berkeley-Stil. Beispielsweise wird das Feld **Content-Length**: ignoriert.

-X *headerfield*
Zeigt den Feldnamen und dessen Inhalt auf einer einzelnen Zeile an.

free

Gibt Informationen über den Hauptspeicherverbrauch aus: Gesamtzahl, benutzte und freie Blöcke, ausgelagerte und mehrfach genutzte Blöcke sowie die benutzten Puffer.

Optionen

-b Berechnet den Speicherverbrauch in Bytes.

-k Berechnet den Speicherverbrauch in Kilobyte (Voreinstellung).

-m Berechnet den Speicherverbrauch in Megabyte.

-o Unterläßt die Angaben zu Puffern bei der Ausgabe.

-s *Sek* Gibt die Informationen alle *Sek* Sekunden aus.

-t Gibt zusätzlich am Ende eine Summe über alle Gesamtgrößen aus.

ftp [*Optionen*] [*Netzwerkrechner*]

Überträgt Dateien zwischen dem lokalen Rechner und einem entfernten *Netzwerkrechner* hin und her. Nach der Optionsliste sind die interaktiven Befehle zur Bedienung des Programms beschrieben. Einige der Befehle sind Schalter, die ein Feature von **ftp** ein- bzw. ausschalten.

Optionen

-d Schaltet die Ausgabe zur Fehlersuche hinzu.

-g Schaltet die Dateinamenserweiterung aus.

-i Schaltet den interaktiven Modus ab.

-n Unterdrückt das automatische Einloggen nach dem Verbindungsaufbau.

-v Ausführliche Meldungen, zeigt alle Antworten der Gegenstelle.

Befehle

![*Programm* [*Argumente*]]
Startet eine interaktive Shell auf der lokalen Maschine. Wenn *Argumente* angegeben sind, wird das erste als Programm ausgeführt, während der Rest der Argumente diesem Programm übergeben wird.

$ *Makro-Name* [*Argumente*]
Führt das mit *Makro-Name* bezeichnete und mit dem Befehl **macdef** erstellte Makro aus. *Argumente* werden unverändert an das Makro übergeben.

account [*Paßwort*]
Liefert ein zusätzliches Paßwort, welches die Gegenstelle nach dem erfolgreichen Login verlangt, um Zugang zu Ressourcen zu

→

gewähren. Wenn kein *Paßwort* zusammen mit dem Befehl angegeben wird, muß das *Paßwort* interaktiv eingegeben werden (ohne Echo).

append *Lokale_Datei* [*Entfernte_Datei*]
Hängt eine lokale Datei an eine Datei auf der entfernten Maschine an. Wird kein Name für die *Entfernte_Datei* angegeben, so dient der lokale Dateiname als Bezeichnung, nachdem er durch eventuelle **ntrans**- oder **nmap**-Einstellungen modifiziert wurde. Für die Dateiübertragung werden die aktuellen Einstellungen von *type*, *format*, *mode* und *structure* verwendet.

ascii Setzt den Transfertyp auf Netzwerk-ASCII (Voreinstellung).

bell Veranlaßt die Ausgabe eines Alarmtons nach dem Abschluß jedes Übertragungsbefehls.

binary Setzt den Transfertyp auf binär.

bye Beendet die FTP-Verbindung und das Programm.

case Schaltet die Umwandlung von Groß-/Kleinschreibung beim Transfer durch **mget** um. Voreinstellung für diesen Schalter ist **aus**. Wenn der Schalter **ein**geschaltet wird, werden vollständig in Großbuchstaben erscheinende Dateinamen von der Gegenstelle in Dateinamen mit Kleinbuchstaben auf dem lokalen Rechner umgewandelt.

cd *Verzeichnis*
Wechselt das Arbeitsverzeichnis auf der Gegenstelle in das angegebene *Verzeichnis*.

cdup Wechselt das Arbeitsverzeichnis auf der Gegenstelle in das Elternverzeichnis.

chmod [*Modus*] [*entfernte_Datei*]
Ändert die Zugriffsrechte der *entfernten Datei*. Wenn kein *Modus* angegeben ist, wird dieser interaktiv abgefragt.

close Beendet eine FTP-Verbindung und kehrt zum Befehlsmodus zurück.

cr Schaltet die CR-Unterdrückung beim ASCII-Dateitransfer um.

delete *Datei*
Löscht die angegebene *Datei* auf der Gegenstelle.

debug [*Modus*]
Schaltet die Ausgabe zur Fehlersuche hinzu. Die optionale Angabe eines *Modus* erlaubt die Feineinstellung.

dir [*Verzeichnis*] [*Datei*]
Gibt eine Liste des Inhalts des *Verzeichnisses* auf der Gegenstelle aus bzw. schreibt es in die optional angegebene lokale *Datei*. Fehlt die Angabe des *Verzeichnisses*, wird das aktuelle Verzeichnis aufgelistet. Wenn keine lokale *Datei* angegeben ist oder an deren Stelle ein – übergeben wird, erfolgt die Ausgabe auf das lokale Terminal.

disconnect
Synonym für **close**.

form *format*
Stellt das Dateitransferformat ein. Voreinstellung ist *file*.

get *entfernte_Datei* [*lokale_Datei*]
Holt die *entfernte_Datei* von der Gegenstelle und speichert sie auf der lokalen Maschine. Wenn die *lokale_Datei* nicht angegeben ist, wird der Name der entfernten Datei verwendet, nachdem er durch die aktuellen Einstellungen von **case**, **ntrans** und **nmap** modifiziert wurde. Wenn anstelle der lokalen Datei ein – angegeben wird, erscheint die Ausgabe auf dem lokalen Terminal.

glob Schaltet die Dateinamenserweiterung für **mdelete**, **mget** und **mput** aus bzw. ein. Wenn **glob** abgeschaltet ist, werden die Dateinamen zeichengetreu übernommen und nicht erweitert.

hash Schaltet die Ausgabe eines Nummernzeichens (#) für jeden übertragenen Datenblock ein bzw. aus.

help [*Befehl*]
Gibt Hilfsinformationen für den angegebenen *Befehl* aus. Ohne Argument wird eine Liste aller verfügbaren Befehle ausgegeben.

idle [*Sekunden*]
Holt/setzt den Leerlaufzeitgeber auf der Gegenstelle. Die Gegenstelle hält eine Verbindung bei Inaktivität für die angegebene Anzahl *Sekunden* offen. Ohne Argument wird die aktuelle Einstellung angezeigt.

image Entspricht dem Befehl **binary**.

lcd [*Verzeichnis*]
Ändert das Arbeitsverzeichnis auf der lokalen Maschine. Wenn kein *Verzeichnis* angegeben ist, wird in das Heimatverzeichnis des Benutzers gewechselt.

ls [*Verzeichnis*] [*Datei*]
Gibt eine Liste des Inhalts des *Verzeichnisses* auf der Gegenstelle aus bzw. schreibt es in die optional angegebene lokale *Datei*. Fehlt die Angabe des *Verzeichnisses*, wird das aktuelle Verzeichnis aufgelistet. Wenn keine lokale *Datei* angegeben ist oder an deren Stelle ein – übergeben wird, erfolgt die Ausgabe auf das lokale Terminal.

macdef *Makroname*
Definiert ein Makro. Aufeinanderfolgende Zeilen werden unter der Bezeichnung *Makroname* gespeichert; eine Leerzeile beendet den Makro-Eingabemodus. Wenn das Symbol **$i** im Makro erscheint, wird dieses bei der Ausführung durch ein Argument ersetzt und die Ausführung wiederholt, bis alle Argumente abgearbeitet sind. Ein zeichengetreues $ kann durch ein \ vor dieser Interpretation geschützt werden.

mdelete *Dateien*
Löscht die angegebenen *Dateien* auf der Gegenstelle.

mdir *Verzeichnisse Datei*
Entspricht dem Befehl **dir**, es können aber mehrere Verzeichnisse angegeben werden.

→

mget *Dateien*
Führt eine Dateinamenserweiterung auf der Gegenstelle durch und überträgt mittels **get** alle auf diese Weise angegebenen *Dateien*.

mkdir *Verzeichnis*
Erzeugt ein neues *Verzeichnis* auf der entfernten Maschine.

mls *entfernte_Dateien lokale_Datei*
Wie **nlist**, es können jedoch mehrere *entfernte_Dateien* angegeben werden, und die Angabe einer lokalen Datei ist erforderlich.

mode [*Modus*]
Setzt den Dateitransfermodus auf den angegebenen Wert. Voreingestellt ist der *stream*-Modus.

modtime [*Datei*]
Zeigt das Datum der letzten Änderung der *Datei* auf der Gegenstelle an.

mput [*Dateien*]
Führt eine Dateinamenserweiterung auf der lokalen Maschine durch und überträgt mittels **put** die auf diese Weise angegebenen *Dateien*.

newer *externe_Datei* [*lokale_Datei*]
Holt die externe Datei, wenn diese neuer ist als die lokale Datei.

nlist [*Verzeichnis*] [*Datei*]
Gibt eine Liste des Inhalts des *Verzeichnisses* auf der Gegenstelle aus bzw. schreibt es in die optional angegebene lokale *Datei*. Fehlt die Angabe des *Verzeichnisses*, wird das aktuelle Verzeichnis aufgelistet.

nmap [*inpattern outpattern*]
Schaltet den Mechanismus zur Dateinamenabbildung ein bzw. aus. Die Abbildung erfolgt entsprechend dem Muster *inpattern*, einer Vorlage für ankommende Dateinamen, und *outpattern*, das den resultierenden Dateinamen bestimmt. Die Symbole $1 bis $9 werden als Variablen behandelt: Beispielsweise würde das *inpattern* **$1.$2** beim eingehenden Dateinamen **readme.txt** die Variable $1 mit **readme** und $2 mit **txt** belegen. Das *outpattern* **$1.data** würde zu einem Ausgabedateinamen **readme.data** führen. $0 enthält den vollständigen Dateinamen. **[string1, string2]** wird durch *string1* ersetzt, sofern dieser nicht leer ist. In diesem Fall würde er duch *string2* ersetzt.

ntrans [*inchars* [*outchars*]]
Schaltet den Mechanismus zur Zeichenübersetzung ein bzw. aus. Entsprechen Buchstaben eines Dateinamens den Zeichen in *inchars*, werden sie durch die entsprechenden Zeichen aus *outchars* ersetzt. Sind keine Argumente angegeben, wird der Mechanismus abgeschaltet. Sind Argumente angegeben, gilt:

- Zeichen in externen Dateinamen werden während der Ausführung der Befehle **mput** und **put** übersetzt, wenn kein Zieldateiname angegeben ist.

- Zeichen in lokalen Dateinamen werden während der Ausführung der Befehle **mget** und **get** übersetzt, wenn kein lokaler Zieldateiname angegeben ist.

open *Host* [*Port*]
: Stellt eine Verbindung zum angegebenen *Host* her. Eine Portnummer kann optional angegeben werden. In diesem Fall versucht **ftp**, die Verbindung über diesen Port aufzubauen.

prompt
: Schaltet die interaktive Eingabeaufforderung für die **m***-Befehle ein bzw. aus.

proxy *ftp-Befehl*
: Führt einen ftp-Befehl auf einer zweiten Steuerverbindung aus, zum Beispiel um Befehle auf zwei verschiedenen Gegenstellen gleichzeitig auszuführen.

put *Datei* [*externe_Datei*]
: Speichert eine lokale Datei auf der Gegenstelle. Wenn kein Name für die *externe_Datei* angegeben ist, wird der lokale Dateiname nach der Umwandlung entsprechend den Einstellungen von **ntrans** und **nmap** verwendet. Der Dateitransfer findet entsprechend den Einstellungen von *type*, *file*, *structure* und *transfer mode* statt.

pwd
: Gibt den Namen des aktuellen Arbeitsverzeichnisses auf der Gegenstelle auf dem lokalen Terminal aus.

quit
: Synonym für **bye**.

quote *arg1 arg2*...
: Sendet die Argumente unverändert an den externen FTP-Server.

recv *Datei* [*lokale_Datei*]
: Synonym für **get**.

reget *Datei* [*lokale_Datei*]
: Holt eine *Datei* (wie **get**), nimmt die Übertragung jedoch am Ende der lokalen Datei wieder auf. Hilfreich, um einen abgebrochenen Transfer fortzusetzen.

remotehelp [*Befehl*]
: Fordert Hilfe über den angegebenen *Befehl* von der Gegenstelle an. Ohne die Angabe eines Befehls wird normalerweise eine Liste der verfügbaren Befehle ausgegeben.

remotestatus [*Datei*]
: Zeigt den Status der Gegenstelle oder, wenn eine *Datei* angegeben ist, den Status dieser Datei an.

rename [*von*] [*nach*]
: Benennt auf der Gegenstelle die Datei *von* um in *nach*.

reset
: Löscht die Antwortwarteschlange.

restart *Bytezahl*
: Nimmt den Dateitransfer an einer bestimmten Bytezahl wieder auf.

→

rmdir [*Verzeichnis*]
Löscht ein *Verzeichnis* auf der entfernten Maschine.

runique
Schaltet den Mechanismus zur lokalen Speicherung mit eindeutigen Dateinamen um. Wenn die Option eingeschaltet ist, werden Dateinamen mit den Endungen .1, .2 usw. ergänzt, wenn das zur Vermeidung von doppelten Dateinamen erforderlich ist. Voreinstellung ist aus.

send *Datei* [*externe_Datei*]
Synonym für **put**.

sendport
Schaltet die Verwendung der **PORT**-Befehle um.

site [*Befehl*]
Holt/setzt systemspezifische Informationan von/auf der Gegenstelle.

size *Datei*
Liefert die Größe der *Datei* von der Gegenstelle.

status Zeigt den aktuellen Status von **ftp** an.

struct [*struct-name*]
Setzt die Dateitransferstruktur entsprechend dem Argument. Als Standard wird die **stream**-Struktur verwendet.

sunique
Schaltet den Mechanismus zur Erzeugung eindeutiger Dateinamen auf der Gegenstelle ein bzw. aus.

system Gibt den Typ des Betriebssystems der Gegenstelle aus.

tenex Setzt den Dateitransfertyp auf den für die Kommunikation mit TENEX-Maschinen erfoderlichen Wert.

trace Schaltet die Paketverfolgung ein bzw. aus.

type [*Typ*]
Setzt den Dateitransfermodus auf den angegebenen Wert. Ist kein Typ angegeben, wird die aktuelle Einstellung angezeigt. Die Voreinstellung ist ASCII.

umask [*Maske*]
Setzt die Maske für den Dateierzeugungsmodus auf der Gegenstelle. Ist keine Maske angegeben, wird der aktuelle Wert angezeigt.

user *Benutzername* [*Paßwort*] [*Account*]
Loggt den Benutzer bei der Gegenstelle ein. **ftp** erfragt Paßwort und Account interaktiv, wenn sie nicht als Argument übergeben wurden.

verbose
Schaltet informative Ausgabe hinzu.

? [*Befehl*]
Entspricht dem Befehl **help**.

fuser [*Optionen*] *Datei* ... [–]

(file user) Gibt zu jeder *Datei* die Prozeß-IDs an, die diese in Gebrauch haben. Den IDs folgen:

c für einen Prozeß, der die Datei als Arbeitsverzeichnis hat.

e für einen Prozeß, der die Datei als Binärcode hat.

f für einen Prozeß, der die Datei mit **open** geöffnet hält.

r für einen Prozeß, zu dem die Datei als Wurzelverzeichnis gehört.

m für einen Prozeß, der die Datei mit **mmap** spiegelt.

Optionen

-	Setzt die Optionen auf ihre Voreinstellungen zurück.
-*Signal*	Versendet *Signal* an Prozesse (voreingestellt: SIGKILL). **fuser** erkennt Angaben als Nummer oder als Name (wie von **fuser -l** angegeben).
-a	Zeigt alle angegebenen Dateien an und nicht nur diejenigen in Gebrauch.
-k	Beendet alle Prozesse, die diese Datei in Gebrauch haben.
-l	Gibt eine Liste der Namen aller Signale aus.
-m	Betrachtet *Datei* als aufgesetztes Dateisysten und macht Angaben zu den Prozessen auf diesem Dateisystem.
-s	Sucht ohne Ausgaben; ignoriert **-a**, **-u** und **-v**.
-u	Gibt zusätzlich die Namen der Prozeßbesitzer aus.
-v	Gibt zusätzliche Angaben aus: PID, Prozeßbesitzer, Befehlsname und Art der Nutzung.

gawk [*Optionen*] *'script' var=Wert* ... *Datei* ...
gawk [*Optionen*] -f *scriptfile var=Wert* ... *Datei* ...

Die GNU-Version von **awk**, ein Programm zur Erkennung von Textmustern, zum Bearbeiten von Textzeilen und für andere Arten der Textmanipulation. Für weitere Informationen, siehe Kapitel 10.

getkeycodes

Gibt die Tabelle des Systemkerns aus, welche die Abbildung von Tastatur-Scancodes zu Keycodes enthält.

grep [*Optionen*] *RegAus* [*Dateien*]

Durchsucht eine oder mehrere *Dateien* nach Zeilen, die dem regulären Ausdruck *RegAus* entsprechen. Eine Beschreibung regulärer Ausdrücke findet sich in Kapitel 5. Der Rückgabewert ist **0**, falls Zeilen ausgegeben wurden, **1**, falls nicht, sowie **2** bei Fehlern. Siehe auch bei **egrep** und **fgrep**.

→

Optionen

- -*num* Gibt *num* Zeilen vor und nach einer Fundstelle aus, es wird jedoch keine Zeile öfter als einmal ausgegeben.
- -b Gibt zu jeder Zeile ihre Blocknummer aus.
- -c (count) Gibt nur die **Anzahl** der gefundenen Zeilen aus.
- -e *RegAus*
  (expression) Sucht nach dem Suchmuster *RegAus*. Diese Option ist sinnvoll für Suchmuster, die mit einem - anfangen.
- -f *AusdrDatei*
  (file) Entnimmt den Suchausdruck der Datei *AusdrDatei*.
- -h Gibt ausschließlich die gefundenen Zeilen aus und nicht die Namen der Dateien.
- -i (ignore case) Behandelt Groß- und Kleinbuchstaben gleich.
- -l Gibt nur **Namen** der Dateien aus, in denen Zeilen gefunden wurden.
- -n Gibt die Zeilennummern zu den gefundenen Zeilen aus.
- -s (silent) Unterläßt Fehlermeldungen über Dateien, die nicht geöffnet werden können.
- -v Gibt alle Zeilen aus, die *nicht* dem Suchmuster entsprechen.
- -A *num*
  (after) Gibt zusätzlich *num* Zeilen vor einer Fundstelle aus.
- -B *num*
  (before) Gibt zusätzlich *num* Zeilen nach einer Fundstelle aus.
- -C (context) Gibt zusätzlich zwei Zeilen vor und nach jeder Fundstelle aus.
- -G (grep) Voreingestellt: Das Suchmuster wird als regulärer Ausdruck behandelt, d.h., die Zeichen ?, +, {, }, |, (und) gelten als normale Zeichen, es sei denn, ihnen ist ein \ vorangestellt.
- -E (egrep) Das Suchmuster ist ein regulärer Ausdruck, in dem die Zeichen ?, +, {, }, |, (und) besondere Bedeutung haben, auch wenn ihnen kein \ vorangestellt ist.
- -F (fgrep) Behandelt das Suchmuster wie **fgrep**.
- -L Gibt die Dateien aus, in denen *keine* Zeilen gefunden wurden.
- -q (quiet) Unterdrückt die normale Ausgabe.
- -w Wendet das Suchmuster nur auf ganze Wörter an. Wörter werden durch Zeichen begrenzt, die weder Buchstabe, Zahl noch der Unterstrich sind.
- -x Wendet das Suchmuster nur auf ganze Zeilen an.

Beispiele

Die Anzahl der Benutzer ausgeben, die die C-Shell nutzen:

```
grep -c /bin/csh /etc/passwd
```

Alle Headerdateien namentlich angeben, die mindestens eine **#include**-Direktive enthalten:

```
grep -l '^#include' /usr/include/*
```

Alle Daten namentlich angeben, die nicht *Suchmuster* enthalten:

```
grep -c Suchmuster files | grep :0
```


gzexe [*Optionen*] [*Datei*]

gzexe komprimiert Programmdateien. Wenn diese gestartet werden, dekomprimieren sie sich selbständig, so daß Plattenplatz auf Kosten von Ausführungszeit eingespart wird. **gzexe** erstellt Backupdateien (*Dateiname~*), die nach Testen der komprimierten Programmdatei gelöscht werden sollten.

Option

-d Dekomprimiere bereits komprimierte Programmdateien.

gzip [*Optionen*] [*Datei(en)*]
gunzip [*Optionen*] [*Datei(en)*]

gzip komprimiert die angegebene(n) Datei(en) (oder die Daten von der Standardeingabe) mit der Lempel-Ziv-Kodierung (LZ77). Die komprimierte Datei wird umbenannt in *Datei.gz*. Dabei werden Zugriffsrechte und Zeitmarken beibehalten und symbolische Links ignoriert. Dekomprimiert wird mit **gunzip**, welches, sofern nicht anders angegeben, dieselben Optionen benutzt wie **gzip**. Dateien, die mit **compress** komprimiert wurden, können ebenfalls dekomprimiert werden.

Optionen

-*Ziffer*, --**fast**, --**best**
: Legt die Kompressionstiefe fest. *Ziffer* kann Werte von 1 bis 9 annehmen, wobei mit steigender Ziffer die Kompressionsrate steigt, die Geschwindigkeit hingegen sinkt. 1 bedeutet also schnellstmögliche Bearbeitung, während 9 die bestmögliche Kompression ergibt.

-c, --**stdout**, --**to-stdout**
: Schreibt die komprimierten Daten auf die Standardausgabe, anstatt die Quelldatei zu ersetzen.

-d, --**decompress**, --**uncompress**
: Entspricht dem Programm **gunzip**.

-f, --**force**
: Erzwingt die Durchführung der Komprimierung. **gzip** fordert normalerweise eine Bestätigung, falls die angegebene Datei mehrere symbolische Links besitzt, die entsprechende **.gz**-Version bereits existiert oder bereits komprimierte Daten von einem Terminal eingelesen werden.

-l, --**list**
: Erwartet bereits komprimierte Dateien als Argument. Die Dateien dürfen mit einer der folgenden Methoden komprimiert sein: **gzip**, **deflate**, **compress**, **lzh** und **pack**. Für jede Datei werden die unkomprimierte und die komprimierte Länge (letztere immer -1 bei Dateien, die nicht von **gzip** stammen),

→

gzip
←

Kompressionsrate und Originalname angegeben. Mit **-v** werden zusätzlich Kompressionsmethode, die 32-bit-CRC der unkomprimierten Daten und die Zeitmarken angezeigt. Mit **-N** werden dabei die in der komprimierten Datei gesicherten Originalnamen und Zeitmarken verwendet.

-q, --**quiet**
: Unterdrückt alle Meldungen.

-r, --**recursive**
: Komprimiert rekursiv die Inhalte eines Verzeichnisses, sofern ein Verzeichnisname als Dateiargument angegeben wurde.

-S *Endung*, --**suffix** *Endung*
: Fügt *Endung* und den Originalnamen an. Vorgabewert ist **gz**. Fehlt beim Dekomprimieren unter Verwendung dieser Option die Angabe einer Endung, so versucht **gunzip**, alle angegebenen Dateien zu dekomprimieren, egal welche Endung sie haben.

-t, --**test**
: Testet komprimierte Dateien auf ihre Fehlerfreiheit.

-v, --**verbose**
: Gibt bei jeder Datei den Namen und die Kompressionsrate an.

-N, --**name**
: (Voreinstellung) Speichert Originalnamen und Zeitmarken in der komprimierten Datei. Bei **gunzip** oder **gzip -d** werden Originalname und Zeitmarken restauriert.

---

head

head [*Optionen*] [*Datei(en)*]

head druckt die ersten Zeilen (Vorgabewert: 10) von einer oder mehreren Dateien. Ohne Angabe von Dateien wird von der Standardeingabe gelesen, bei mehr als einer Datei wird eine Kopfzeile für jede Datei ausgegeben.

Optionen

-c *Anzahl*, --**bytes** *Anzahl*
: Druckt die angegebene *Anzahl* oder, falls nach *Anzahl* ein **k** oder **m** steht, die ersten *Anzahl* Kilobytes oder Megabytes.

-l, -n *Anzahl*, --**lines** *Anzahl*
: Druckt die angegebene *Anzahl* Zeilen. Vorgabewert ist 10.

-v, --**verbose**
: Druckt auch für nur eine Datei eine Kopfzeile.

Beispiele

Zeigt die ersten 20 Zeilen der Datei **telefonliste**:

```
head -20 telefonliste
```

Zeigt die ersten zehn Telefonnummern mit der Vorwahl 030:

```
grep '(030)' telefonliste | head
```

host [*Optionen*] *Host* [*Server*]
host [*Optionen*] *Zone* [*Server*]

host liefert Informationen über bestimmte Hosts oder Zonen im DNS. Für *Host* kann IP-Adresse oder Hostname angegeben werden; **host** konvertiert standardmäßig IP-Adressen in Hostnamen und fügt bei Hosts ohne führenden Punkt die Domain hinzu. Vorgabeserver werden aus */etc/resolv.conf* ermittelt. Für weitere Informationen über Hosts und Zonen wird auf die Kapitel 1 und 2 des Buches *DNS und BIND* von Paul Albitz und Cricket Liu verwiesen.

Optionen

-a
: Entspricht **-t ANY**.

-c *Klasse*
: Sucht nach der entsprechenden *Resource Record Klasse* (IN, INTERNET, CS, CSNET, CH, CHAOS, HS, HESIOD, ANY oder *). Vorgabewert ist IN.

-d
: Schaltet den Debugging-Modus ein. **-dd** ist die ausführlichere Version.

-e
: Unterdrückt Informationen über Domains außerhalb der angegebenen Zone. Bei Anfragen über Hostnamen wird weder „additional information" noch „authoritative nameserver" ausgegeben.

-f *Datei*
: Leitet alle Ausgaben sowohl in *Datei* als auch auf die Standardausgabe.

-i
: Ermittelt aus einer IP-Adresse die zugehörige *in-addr.arpa*-Adresse, die Klasse (immer PTR) und den Hostnamen.

-l *Zone*
: Zeigt alle Hosts in *Zone* an.

-m
: Listet ausschließlich MR-, MG- und MB-Einträge auf. Dabei werden MR- (renamed mail box = umbenannter Briefkasten) und MG- (mail group = Post-Gruppe) Einträge rekursiv in die entsprechenden MB- (mail box = Briefkasten) Einträge aufgeschlüsselt.

-o
: Unterdrückt die Ausgabe von Texten auf die Standardausgabe.

-p [*Server*]
: Fragt in Verbindung mit **-l** nur den primären Nameserver (oder *Server*) einer Zone für Übertragungen innerhalb einer Zone ab und nicht die eventuell antwortenden Authoritative Nameservers. Nützlich zum Testen von unregistrierten Zonen.

-q
: Unterdrückt Warnungen (quiet). Fehlermeldungen werden trotzdem ausgegeben.

-r
: Fordert den angesprochenen Server auf, keine anderen Server für die Anfrage hinzuzuziehen. Es sollen ausschließlich die von ihm selbst gespeicherten Informationen benutzt werden.

-t *Typ*
: Sucht nach Einträgen vom Typ *Typ* im Resource Record. Für *Typ* kann A, NS, PTR, ANY oder * (all) verwendet werden.

→

-u Verwendet TCP und nicht UDP.

-v Listet alle Felder des Resource Record auf (verbose), auch „Lebensdauer", Klasse, „additional information" und „authoritative nameservers", soweit vom angesprochenen Nameserver bereitgestellt.

-vv Fügt außerdem noch Informationen über die Voreinstellungen des *host* hinzu (very verbose).

-w Wartet ohne Timeout auf den angesprochenen Server.

-x Erlaubt es, mehrere Hosts oder Zonen anzugeben. Wird auch ein Server angegeben, muß dem Argument ein -X vorausgehen.

-A Sucht bei Angabe eines Hostnamens nach der zugehörigen IP-Adresse und dann rückwärts nach dem zur IP-Adresse gehörenden Hostnamen, um die Übereinstimmung zu prüfen. Wird eine IP-Adresse angegeben, so wird nach dem zugehörigen Hostnamen gesucht und überprüft, ob der Host auf diesen Namen reagiert. Bei Angabe einer Zone werden die IP-Adressen aller Hosts überprüft. Werden keine Unstimmigkeiten gefunden, so wird keine Meldung ausgegeben.

-C Arbeitet ähnlich wie -l, jedoch wird überprüft, ob die Nameserver der Zone wirklich „authoritative" sind. Die SOA (start of authority) Records einer Zone weisen „Authoritative Nameservers" im NS-Feld aus. Nur solche Server werden abgefragt. Zusätzlich werden weitere Überprüfungen durchgeführt.

-D Arbeitet ähnlich wie -H, bearbeitet jedoch auch Hosts mit mehr als einer Adresse je Name.

-E Arbeitet ähnlich wie -H, meldet jedoch Hosts in einer undefinierten Subdomain nicht als Fehler.

-H *Zone*
Gibt die Anzahl der verschiedenen Hosts in *Zone* aus. Aliase werden nicht mitgezählt. Außerdem werden Doppelvergaben und Hosts in undefinierten Subdomains als Fehler aufgelistet.

-F *Datei*
Lenkt die Standardausgabe nach *Datei* um und gibt zusätzliche Resource- Record-Angaben über die Standardausgabe aus.

-G *Zone*
Arbeitet ähnlich wie -H, zählt jedoch Gateway Hosts mit.

-I *Zeichen*
Unterdrückt Warnungen bei Domain-Namen mit illegalen Zeichen *Zeichen*, wie etwa '_'.

-L *Level*
Listet bei Gebrauch mit -l rekursiv alle deligierten Zonen in der angegebenen Zone mit einer Schachtelungstiefe von maximal *Level* auf.

-P *Server*
Gibt bei Verwendung mit -l die vorzugsweise zu benutzenden Sekundärserver bei der Übertragung der Zonendaten an. *Server* ist eine durch Kommata getrennte Liste. Die höchste Priorität

erhalten die Server, die mit den meisten Domainkomponenten in dem jeweiligen Abschnit von *Server* übereinstimmen.

-R
: Behandelt nicht vollständig angegebene Hostnamen so, wie dies auch von BIND getan wird. Dabei wird jede Komponente der lokalen Domain untersucht.

-S
: Gibt, wenn zusammen mit -l verwendet, alle Hosts in der Zone über die Standardausgabe aus. Dabei werden Klasse und IP-Adresse mit ausgegeben, Hosts in Unterzonen jedoch ignoriert. Warnmeldungen (illegale Namen, fehlende Records etc.) werden über den Standardfehlerkanal ausgegeben.

-T
: Gibt die „Lebensdauer" der Einträge aus, also wie lange die Informationen über einen Host noch gespeichert werden, bevor der Nameserver sie auffrischt.

-X *Server*
: Gibt an, welcher Server angesprochen werden soll, und erlaubt die Angabe mehrerer Hosts oder Zonen.

-Z
: Fügt bei der Ausgabe von Resource Records einen einleitenden Punkt bei Domain-Namen hinzu und gibt „Lebensdauer" und Klasse mit an.

hostname [*Option*] [*Hostname*]

hostname setzt oder liefert den Namen des lokalen Hosts. Nur privilegierte Benutzer können den Namen mit dem *Hostname*-Argument setzen.

Option

-d, --domain
: Liefert den DNS Domain-Namen.

-f, --fqdn, --long
: Liefert den vollständigen Domain-Namen.

-s, --short
: Entfernt die Domaininformationen vom ausgegebenen Namen.

-F *Datei*, **--file** *Datei*
: Entnimmt den Hostnamen aus *Datei*.

id [*Optionen*] [*Benutzername*]

id zeigt Informationen über den aktuellen oder angegebenen Benutzer an: BenutzerID, GruppenID, effektive Benutzer- und GruppenID, falls von Bedeutung, sowie zusätzliche GruppenIDs.

Optionen

-g, --group
: Gibt lediglich die GruppenID aus.

-n, --name
: Liefert zusammen mit **-u**, **-g** oder **-G** den Benutzer- oder Gruppennamen anstatt der ID-Nummer.

→

id ←

-r, --real
: Liefert zusammen mit -u, -g oder -G, die reale BenutzerID oder GruppenID anstatt der effektiven IDs.

-u, --Benutzer
: Gibt nur die BenutzerID aus.

-G, --groups
: Liefert nur zusätzliche GruppenIDs.

---

info

info [*Optionen*] *Stichwörter*

info ist ein GNU-Hypertext-Betrachter: Er zeigt Onlinedokumentationen an, die aus Texinfo-Eingaben hergestellt wurden. Infodateien besitzen eine hierarchische Anordnung und können Menüs für Unterbegriffe enthalten. Wird **info** ohne Angabe von Optionen gestartet, so wird zunächst die Hauptinfodatei angezeigt (üblicherweise */usr/info/dir*). Wenn mehrere *Stichwörter* angegeben sind, so wird das erste Stichwort aus dem Menü der Hauptinfodatei herausgesucht, das nächste Stichwort in dem Menü der dadurch bestimmten Unterdatei usw. Die Anzeige zu Beginn kann außerdem durch die -f- und -n- Optionen beeinflußt werden.

Optionen

-d *Verzeichnisse*, --**directory** *Verzeichnisse*
: Durchsucht *Verzeichnisse*, eine durch Kommata getrennte Liste, nach Infodateien. Wird diese Option nicht verwendet, so wird die INFOPATH Environment-Variable oder der Vorgabewert */usr/info* verwendet.

--**dribble** *Datei*
: Speichert jeden Tastendruck in der angegebenen *Datei*. Die gespeicherten Tastendrücke können später mit der --**restore**-Option benutzt werden, um schnell wieder an dieselbe Information zu kommen.

-f *Datei*, --**file** *Datei*
: Zeigt eine bestimmte Infodatei an.

-n *node*, --**node** *node*
: Zeigt den angegebenen Knoten in der Infodatei an.

-o *Datei*, --**output** *Datei*
: Schreibt die Informationen in *Datei* anstatt auf den Bildschirm.

--**help** Zeigt eine kurze Hilfe an.

--**restore** *Datei*
: Führt nach dem Start die in *Datei* gespeicherten Tastendrücke aus.

--**subnodes**
: Zeigt die in der Infodatei verwalteten Unterbegriffe an.

--**version**
: Gibt die Versionsnummer des Programms aus.

ispell [*Optionen*] [*Dateien*]

ispell vergleicht die Wörter in einer oder mehreren angegebenen *Dateien* mit dem Systemwörterbuch. Unbekannte Wörter werden am oberen Bildschirmrand angezeigt, zusammen mit der wahrscheinlich korrekten Schreibweise. Eine Textkorrektur ist mittels verschiedener Befehle möglich.

Optionen

-b Legt eine Sicherheitskopie der Datei in *Datei*.**bak** an.

-d *Datei*
Benutzt *Datei* statt des Systemwörterbuches.

-m Schlägt verschiedene Deklinations- und Konjugationsformen vor.

-n Interpretiert die angegebene Datei als nroff- oder troff- Datei.

-p *Datei*
Benutzt *Datei* anstatt des persönlichen Wörterbuches.

-t Interpretiert die angegebene Datei als TEX oder LATEX Datei.

-w *Zeichen*
Definiert *Zeichen* als erlaubt, zusätzlich zu a–z und A–Z.

-x Legt keine Sicherheitskopie der Originaldatei an.

-B Untersucht unbekannte Wörter zusätzlich darauf, ob sie aus zwei bekannten Wörtern bestehen, zwischen denen lediglich ein Leerzeichen fehlt.

-C Unterdrückt Fehlermeldungen, wenn lediglich ein Leerzeichen fehlt.

-L *Anzahl*
Zeigt *Anzahl* Zeilen aus dem laufenden Text an, um den Zusammenhang darzustellen.

-M Listet interaktive Befehle am unteren Bildschirmrand auf.

-N Unterdrückt die Anzeige der interaktiven Befehle.

-P Unterdrückt den Vorschlag von unterschiedlichen Deklinations- und Konjugationsformen.

-S Sortiert Vorschläge nach ihrer Wahrscheinlichkeit.

-T *Typ*
Interpretiert alle Dateien als mit *Typ* formatiert.

-W *n* Ignoriert Wörter mit *n* oder weniger Buchstaben.

-V Benutzt 'Hut'-Schreibweise (^L) für die Anzeige von Steuerzeichen und **M-** für die Anzeige von Zeichen mit gesetztem höchstwertigem Bit.

Interaktive Befehle

Nummer
Ersetzt falsches Wort mit Vorschlag Nummer *Nummer*.

!*Kommando*
Ruft die Shell auf und führt das *Kommando* aus. Vor der Rückkehr wird auf einen Tastendruck gewartet.

a Akzeptiert das Wort als korrekt, nimmt es jedoch nicht in das persönliche Wörterbuch auf.

→

ispell ←	i — Fügt das Wort dem persönlichen Wörterbuch hinzu. l — Durchsucht das Systemwörterbuch. q — Beendet das Programm, ohne die Änderungen zu speichern. r — Ersetzt das falsche Wort. u — Fügt das Wort in Kleinbuchstaben dem persönlichen Wörterbuch hinzu. x — Speichert die Änderungen und springt zur nächsten Datei.

kbd_mode

kbd_mode [*Option*]

kbd_mode gibt den aktuellen Tastaturmodus aus. Dies kann RAW, MEDIUMRAW oder XLATE sein.

Optionen

- -a Setzt Modus auf XLATE (ASCII-Modus).
- -k Setzt Modus auf MEDIUMRAW (Tastaturcode-Modus).
- -s Setzt Modus auf RAW (Scancode-Modus).
- -u Setzt Modus auf UNICODE (UTF-8-Modus).

kill

kill [*Option*] *IDs*

kill sendet ein Signal, um einen oder mehrere Prozesse abzubrechen. Der Benutzer benötigt das Zugriffsrecht auf die genannten Prozesse. Dieses Programm arbeitet ähnlich wie der **kill**-Befehl, der in die Bourne, Korn oder C-Shells eingebaut ist. Ein Minuszeichen vor einer *ID* definiert eine ProzeßgruppenID. (Die eingebaute Version unterstützt keine ProzeßgruppenIDs, erlaubt dafür aber Job-IDs, die mit % beginnen.)

Optionen

- -l Listet alle Signale auf.
- -*Signal* Definiert die Signalnummer (von **ps -f**) oder den Signalnamen (von **kill -l**). Mit einer Signalnummer von 9 kann der Abbruch nicht abgefangen werden. Das Vorgabesignal ist TERM.

killall

killall [*Optionen*] [*Name*]

killall beendet Prozesse anhand ihres Namens. Führt mehr als ein Prozeß das angegebene Kommando aus, so werden alle beendet. Namen, die ein / enthalten, werden als Programmdateien interpretiert. Es werden alle Prozesse beendet, die dieses Programm ausführen.

Optionen

- -*Signal* Sendet *Signal* und den Prozeß (Vorgabewert ist TERM). *Signal* kann ein Signalname oder eine Nummer sein.
- -i Fragt nach, bevor ein Prozeß abgebrochen wird.
- -l Gibt eine Liste aller verfügbaren Signale aus.
- -v Zeigt nach dem Löschen eines Prozesses eine Erfolgsmeldung mit der ProzeßID an (verbose).

killall5

killall5

killall5 ist das System-V-Equivalent von **killall**. Dieses Programm löscht alle laufenden Prozesse bis auf diejenigen, von denen es selber abhängt.

less [*Optionen*] [*Dateiname(n)*]

less

less ist ein Programm zum Durchblättern von Dateien oder Bildschirmausgaben. Es wurde geschrieben, um die mangelnde Funktionalität von **more** auszugleichen. Einige der Befehle erlauben eine vorherige Zifferneingabe.

Optionen

-[z]*Anzahl*
: Setzt die Anzahl der jeweils anzuzeigenden Zeilen auf den angegebenen Wert. Vorgabewert ist eine Bildschirmseite. Eine negative *Anzahl* setzt die Anzahl der Zeilen auf *Anzahl* abzüglich der momentanen Zeilenzahl.

+[+]*Befehl*
: Führt den angegebenen *Befehl* gleich beim Start aus. Falls der *Befehl* eine Zahl ist, wird zu der entsprechenden Zeile gesprungen. Die Option ++ läßt diesen Befehl auf jede Datei in der Dateiliste der Kommandozeile wirken.

-? Zeigt einen Hilfebildschirm. Alle anderen Optionen werden ignoriert, es werden auch keine Dateiinhalte angezeigt.

-a Bewirkt, daß bei Suchbefehlen ab der letzten angezeigten Zeile gesucht wird, nicht von der zweiten angezeigten Zeile an.

-b*Anzahl*
: Benutzt die angegebene *Anzahl* Puffer für jede Datei (Vorgabe: 10). Puffer belegen jeweils 1 KB Speicher.

-c Führt einen Bildschirmaufbau von oben, nicht von unten her durch.

-d Unterdrückt die Anzeige von Terminal-Fehlermeldungen.

-e Beendet das Programm, nachdem zweimal *EOF* erreicht wurde.

-f Unterdrückt Warnungen beim Öffnen von Binärdateien.

-g Zeigt nur den vom letzten Suchkommando gefundenen Text hervorgehoben an, nicht alle passenden Textstellen.

-h*Anzahl*
: Blättert niemals mehr als *Anzahl* Zeilen auf einmal zurück.

-i Macht Suchanfragen unabhängig von Groß-/Kleinschreibung, solange der zu suchende Text keine Großbuchstaben enthält.

-j*Nummer*
: Positioniert die „Zielzeile" auf der *Nummer*ten Bildschirmzeile. „Zielzeile" kann das Ergebnis einer Suche oder eines Sprunges sein. Die Zeilen werden vom oberen Bildschirmrand aus gezählt. Eine negative *Nummer* wird vom unteren Bildschirmrand aus gezählt.

-k*Datei*
: Liest *Datei* zur Definition einer speziellen Tastaturbelegung.

→

-m Zeigt eine dem **more**-Programm entsprechende Statusmeldung (Prompt) inklusive der Prozentanzeige des bereits angezeigten Textes an.

-n Schaltet das Zählen der Zeilennummern ab. Dadurch werden die -**m**- und -**M**-Optionen sowie die Befehle = und **v** beeinflußt (die Zeilennummer wird nicht an den Editor weitergegeben).

-o*Datei*

Kopiert die Ausgabe von aus einer Pipe stammenden Informationen in *Datei* und auf den Bildschirm. Es wird nach einer Bestätigung verlangt, wenn *Datei* bereits existiert.

-p*Text* Sucht beim Start gleich nach dem ersten Auftreten von *Text*.

m	Setzt ein kurzes Prompt (definiert durch -**m**).
M	Setzt ein langes Prompt (definiert durch -**M**).
=	Setzt eine Nachricht entsprechend dem Befehl =.

-q Deaktiviert den Signalton bei Versuchen, vor den Dateianfang oder hinter ein *EOF* zu blättern. Statt dessen wird, wenn möglich, ein optisches Signal erzeugt.

-r Zeigt Steuerzeichen direkt an, anstatt die übliche ^*x*-Notation zu benutzen. Diese Option kann bei Erscheinen bestimmter Steuerzeichen zu Anzeigeproblemen führen.

-s Druckt aufeinanderfolgende Leerzeilen als eine Zeile.

-t*Tag* Editiert die Datei, die *Tag* enthält. Es wird in der Datei **./tags** nachgeschlagen (angelegt von **ctags**).

-u Behandelt Backspaces und Carriage Returns als druckbare Zeichen.

-w Zeigt Zeilen nach einem *EOF* als Leerzeilen an, anstatt sie mit einer Tilde (~) zu füllen.

-x*n* Setzt die Tabulatorweite auf *n* Zeichen. Vorgabewert ist 8.

-y*n* Blättert niemals mehr als *Anzahl* Zeilen auf einmal vorwärts.

-B Verhindert das automatische Belegen von Puffern für Daten aus einer Pipe. Falls durch -**b** eine Pufferzahl vorgegeben wurde, werden genau so viele Puffer belegt. Falls notwendig, werden Informationen aus vorhergehenden Bildschirmen gelöscht, um Platz zu schaffen.

-C Zeichnet den Bildschirm neu, indem er gelöscht und dann von oben herab aufgebaut wird.

-E Beendet das Programm, nachdem *EOF* erreicht wurde.

-G Unterdrückt das Hervorheben gefundener Textstellen.

-I Macht Suchanfragen unabhängig von Groß-/Kleinschreibung, auch wenn der zu suchende Text Großbuchstaben enthält.

-M Zeigt noch mehr Informationen im Prompt als mit -**m** inklusive Prozentsatz der angezeigten Daten, Gesamtzeilenzahl und aktueller Zeilennummer.

-N Druckt die aktuelle Zeilennummer vor jeder Zeile.

-O*Datei*

Entspricht -o, aber überschreibt vorhandene Dateien ohne Rückfrage.

-P[m,M,=]*Prompt*
: Setzt das *Prompt* (entsprechend **-m**, **-M** oder =). Vorgegeben ist ein kurzes Prompt (**-m**).

-Q
: Unterdrückt die Auslösung eines Signaltones.

-S
: Schneidet lange Zeilen ab, anstatt sie auf die nächste Zeile umzubrechen.

-T*Datei*
: Liest zusammen mit der **-t** Option oder dem **:t** Befehl statt

-U
: Interpretiert Backspaces und Carriage Returns als Steuerzeichen.

-X
: Unterdrückt die Initialisierungs- und Deinitialisierungssequenzen von Termcap.

Befehle

Einigen Befehlen können Argumente vorangestellt werden.

SPACE, ^V, f, ^F
: Blättert um die eingestellte Zeilenzahl weiter (üblicherweise eine Bildschirmseite).

z
: Wirkt ähnlich wie SPACE, jedoch kann hier die gewünschte Zeilenzahl mit angegeben werden, die dann auch für SPACE gilt.

RETURN, ^N, e, ^E, j, ^J
: Blättert um die eingestellte Zeilenzahl weiter (üblicherweise eine Zeile), auch wenn dies mehr als eine Bildschirmseite wäre.

d, ^D
: Blättert um eine halbe Bildschirmseite weiter. Wird ein Wert angegeben, so wird dieses als neue Zeilenzahl festgelegt.

b, ^B, ESC-v
: Blättert rückwärts, normalerweise eine Bildschirmseite weit.

w
: Entspricht **b**, erlaubt aber das Einstellen eines Standardwertes, der dann auch für **b** gilt.

y, ^Y, ^P, k, ^K
: Blättert rückwärts. Voreingestellt ist eine Zeile. Bei Angabe einer größeren Zeilenzahl wird auch dann geblättert, wenn dabei eine Bildschirmseite überschritten wird.

u, ^U
: Blättert rückwärts, üblicherweise eine halbe Bildschirmseite weit. Wird eine Zeilenzahl angegeben, so ist diese der neue Vorgabewert.

r, ^R, ^L
: Baut den Bildschirm neu auf.

R
: Entspricht **r**, löscht jedoch bereits gepufferte Eingaben.

F
: Blättert vorwärts. Wenn dabei ein *EOF* erreicht wird, so wird versucht, weitere Daten zu lesen. **less** verhält sich dann ähnlich wie **tail -f**.

g, <, ESC-<
: Springt zu einer Zeile. Voreingestellt ist Zeile 1.

G, >, ESC->
: Springt zu einer Zeile. Voreingestellt ist die letzte Zeile.

p, %
: Fragt nach einer Zahl zwischen 0 und 100. Anschließend wird die Stelle angesprungen, die dem angegebenen Prozentwert des Gesamttextes entspricht.

→

less
←

{ Sucht die passende }, falls die erste Bildschirmzeile eine { enthält. Sind mehrere { in der ersten Bildschirmzeile, wird nach der Nummer gefragt, deren Gegenstück gesucht werden soll.

} Sucht die passende {, falls die letzte Bildschirmzeile eine } enthält. Befinden sich mehrere } in der untersten Bildschirmzeile, so wird nach der Nummer gefragt, deren Gegenstück gesucht werden soll.

(Entspricht {, jedoch für (.

) Entspricht }, jedoch für).

[Entspricht {, jedoch für [.

] Entspricht }, jedoch für].

ESC-^F Entspricht {, fragt jedoch nach zwei Zeichen, auf die sich die Suche beziehen soll.

ESC-^B Entspricht }, fragt jedoch nach zwei Zeichen, auf die sich die Suche beziehen soll.

m Fragt nach einem Kleinbuchstaben, der als Markierung für die aktuelle Zeile benutzt werden soll.

' Fragt nach einem Kleinbuchstaben und springt die durch ihn markierte Zeile an. Folgende Sonderzeichen sind vordefiniert:

- ' Kehrt zur Stelle vor der letzten 'großen Bewegung' zurück.
- ^ Springt an den Dateianfang.
- $ Springt an das Dateiende.

^X^X Entspricht '.

/*Zeichenkette*

Sucht nach dem nächsten Auftauchen von *Zeichenkette*, ausgehend von der zweiten angezeigten Zeile. Vor *Zeichenkette* können folgende Sonderzeichen angewendet werden:

- ! Sucht nach Zeilen, die *Zeichenkette* NICHT enthalten.
- * Sucht in der nächsten Datei weiter, falls *Zeichenkette* in der aktuellen Datei nicht vorkommt.
- @ Sucht von der ersten Zeile der ersten angegebenen Datei an, egal, wo man sich gerade befindet.

?*Zeichenkette*

Sucht rückwärts, anfangend bei der Zeile vor der ersten angezeigten Zeile. !, * und @ werden wie bei / interpretiert.

ESC-/*Zeichenkette*

Entspricht /* .

ESC-?*Zeichenkette*

Entspricht ?* .

n Wiederholt die letzte Suche nach *Zeichenkette*.

N Wiederholt die letzte Suche nach *Zeichenkette*, jedoch in umgekehrter Richtung.

ESC-n Wiederholt die letzte Suche, jedoch als wäre sie mit * versehen gewesen.

ESC-N Wiederholt die letzte Suche, jedoch als wäre sie mit * versehen gewesen, und in umgekehrter Richtung.

ESC-u Schaltet das Hervorheben gefundener Textstellen ein und aus.

:e [*Dateiname*]
: Liest *Dateiname* ein und fügt die Datei der Liste der zu bearbeitenden Dateien hinzu. Fehlt die Angabe von *Datei*, wird die aktuelle Datei neu eingelesen. *Dateiname* kann Sonderzeichen enthalten:

% Aktueller Dateiname

Name der vorherigen Datei

^X^V, E
: Entspricht :e.

:n Liest die nächste Datei in der Dateienliste.

:p Liest die vorherige Datei in der Dateienliste.

:x Liest die erste Datei in der Dateienliste.

:f, =, ^G
: Gibt folgende Informationen aus: Dateiname, Position in der Dateienliste, Zeilennummer der ersten Bildschirmzeile, Gesamtanzahl der Zeilen, Zeichennummer des ersten Zeichens auf dem Bildschirm und die Gesamtzeichenzahl.

- Führt, gefolgt von einer gültigen Kommandozeilenoption, die entsprechende Einstellung am laufenden Programm durch. Ein eventuell erforderliches Argument wird interaktiv erfragt.

-+ Setzt, gefolgt von einer gültigen Kommandozeilenoption, die entsprechene Einstellung auf den Standardwert zurück.

-- Setzt, gefolgt von einer gültigen Kommandozeilenoption, die entsprechene Einstellung auf das Gegenteil vom Standardwert, wenn solch ein Wert eindeutig bestimmt ist.

_ Zeigt, gefolgt von einer gültigen Kommandozeilenoption, die aktuelle Einstellung an.

+*Befehl*
: Führt den *Befehl* jedesmal aus, wenn eine neue Datei eingelesen wird.

q, :q, :Q, ZZ
: Beendet das Programm.

v Ist nicht in allen Versionen verfügbar. Es wird **vi** bzw. der in $VISUAL oder $EDITOR angegebene Editor aufgerufen.

! *Kommando*
: Ist nicht in allen Versionen verfügbar. Es wird **sh** bzw. $SHELL aufgerufen. Wurde ein *Kommando* angegeben, wird es ausgeführt und die Shell beendet. Folgende Sonderzeichen sind verfügbar:

% Name der aktuellen Datei

Name der letzten Datei

!! Letzter Shell-Befehl

→

| <*Markierung*> *Kommando*

Ist nicht in allen Versionen verfügbar. Ein Ausschnitt der Datei, beginnend mit der ersten Bildschirmzeile und endend bei *Markierung*, wird an das *Kommando* weitergereicht. *Markierung* kann eine gesetzte Markierung oder eines der folgenden Sonderzeichen sein:

^	Ab Dateianfang
$	Bis Dateiende
., newline	Aktueller Bildschirminhalt

Statusmeldungen

Die Statusmeldung (Prompt) kann über die Option **-P** frei definiert werden. Dabei werden bestimmte Zeichenketten besonders interpretiert. Diejenigen, die mit % anfangen, werden immer ausgewertet, die mit ? beginnenden Ausdrücke werden nur ausgewertet, wenn bestimmte Bedingungen zutreffen.

Einige Audrücke bestimmen die Position bestimmter Bildschirmzeilen. Diese Ausdrücke benötigen eine Angabe über die anzuwendende Methode. Es gibt fünf verschiedene Zeilen, auf die sich diese Ausdrücke beziehen können:

t	Oberste Bildschirmzeile
m	Bildschirmmitte
b	Unterer Bildschirmrand
B	Die nächste Zeile unterhalb des sichtbaren Bereiches
j	„Zielzeile"

%b{t,m,b,B,j}
: Ermittelt die Anzahl der Bytes seit Dateibeginn.

%B — Ergibt die Größe der aktuellen Datei in Bytes.

%E — Liefert den Namen des Editorprogramms.

%f — Liefert den Namen der aktuellen Datei.

%i — Liefert die Position der aktuellen Datei in der Dateienliste.

%l{t,m,b,B,j}
: Ermittelt die Zeilennummer einer bestimmten Zeile.

%L — Bestimmt die letzte Zeilennummer der aktuellen Datei.

%m — Bestimmt die Gesamtzahl aller Zeilen in allen Dateien der Dateienliste.

%p{t,m,b,B,j}
: Ermittelt den Prozentsatz des bereits angezeigten Textes.

%s — Entspricht %B.

%t — Entfernt angehängte Leerzeichen.

%x — Liefert die nächste Datei in der Dateienliste.

?a — Wahr, wenn die Statuszeile bereits Text enthält.

?b{t,m,b,B,j}
: Wahr, wenn die Byteposition ermittelt werden kann.

?B
: Wahr, wenn die Dateilänge der aktuellen Datei ermittelt werden kann.

?e
: Wahr, wenn *EOF* erreicht wurde.

?f
: Wahr, wenn von einer Datei und nicht von einer Pipe gelesen wird.

?l{t,m,b,B,j}
: Wahr, wenn die Zeilennummer ermittelt werden kann.

?L
: Wahr, wenn die letzte Zeilennummer ermittelt werden kann.

?m
: Wahr, wenn mehrere Dateien in der Dateinamenliste stehen.

?n
: Wahr, wenn dies die erste Statusmeldung der aktuellen Datei ist.

?p{t,m,b,B,j}
: Wahr, wenn der Prozentsatz der bereits angezeigten Information ermittelt werden kann.

?s
: Entspricht ?B.

?x
: Wahr, wenn die aktuelle Datei nicht die letzte ist.

---

ln [*Optionen*] *Quellname* [*Zielname*]
ln [*Optionen*] *Quellnamen Zielverzeichnis*

ln erstellt Pseudonyme (Verknüpfungen) für Dateien, so daß diese unter verschiedenen Namen angesprochen werden können. In der ersten Form wird die Datei *Quellname* mit dem Pseudonym *Zielname* verknüpft, wobei *Zielname* normalerweise ein neuer Dateiname oder (falls nicht angegeben) der Name des aktuellen Verzeichnisses ist. Ist *Zielname* eine existierende Datei, so wird sie mit der Verknüpfung auf *Quellname* überschrieben; ist *Zielname* ein existierendes Verzeichnis, wird eine Verknüpfung namens *Quellname* auf dieses Verzeichnis angelegt. In der zweiten Form werden im *Zielverzeichnis* gleichnamige Verknüpfungen auf die *Quellnamen* angelegt.

Optionen

-b, --backup
: Kopiert Dateien, bevor sie gelöscht werden.

-d, -F, --directory
: Soll feste Verknüpfungen auf Verzeichnisse erzeugen. Um die Konsistenz des Dateisystems zu erhalten, ist diese Funktion unter Linux verboten.

-f
: Unterdrückt die Rückfrage bei Überschreiben von Dateien (force).

-i, --interactive
: Fragt nach Erlaubnis, bevor Dateien überschrieben werden.

-n, --no-dereference
: Ersetzt symbolische Verknüpfungen anstatt sie als Referenz auf ihre Verknüpfung auszuwerten. --force ist nützlich in diesem Zusammenhang.

→

ln ←

-s
: Erstellt eine symbolische Verknüpfung. Dies erlaubt Verknüpfungen zu Verzeichnissen und über Dateisystemgrenzen hinweg und ermöglicht es, den Namen der verknüpften Datei zu sehen, wenn **ls -l** aufgerufen wird (andernfalls wäre der Name des Verknüpfungszieles unsichtbar).

-S *Endung*, --**suffix** *Endung*
: Hängt *Endung* an die Sicherheitskopien an anstatt der Vorgabeendung ~.

-V, --**version-control {numbered,existing,simple}**
: Definiert die Namensgebung beim Anlegen von Sicherheitskopien.

t, numbered	Numeriert.
nil, existing	Einfach (~), es sei denn, eine numerierte Sicherheitskopie existiert bereits. In diesem Falle wird eine weitere numerierte Kopie angelegt.
never, simple	Einfach.

locate

locate [*Optionen*] *Zeichenkette*

locate durchsucht Datenbanken mit Dateinamen und listet Übereinstimmungen auf. *, ?, [und] werden gesondert behandelt; / und . nicht. Als Übereinstimmung gewertet werden alle Dateien, die *Zeichenkette* enthalten, es sei denn, *Zeichenkette* enthält eines oder mehrere der oben aufgeführten Sonderzeichen. In diesem Falle muß die Übereinstimmung vollständig sein.

Option

-d *Pfad*, --**database**=*Pfad*
: Durchsucht Datenbanken in *Pfad*. *Pfad* muß eine durch Kommata getrennte Liste sein.

lockfile

lockfile [*Optionen*] *Dateiname(n)*

lockfile generiert Signaldateien, um Zugriffe auf eine Datei zu begrenzen. Wenn eine Signaldatei nicht angelegt werden kann (da schon vorhanden), so wird eine Pause von acht Sekunden eingelegt und dann ein erneuter Versuch gestartet, bis es gelingt. Das Programm bearbeitet Flags in der Reihenfolge der Angabe. Das bedeutet, daß ein Flag nur die nach ihm angegebenen Dateien betrifft.

Optionen

-!
: Invertiert den Rückgabewert.

-r *Anzahl*
: Bricht nach der angegebenen *Anzahl* vergeblicher Versuche ab. Vorgabewert ist -1 (unendlich). Wird abgebrochen, werden alle bereits angelegten Signaldateien wieder gelöscht.

login [*Name* | *Option*]

login

login meldet einen Benutzer bei einem System an. Der Benutzer wird nach seinem Benutzernamen (*Name* kann in der Kommandozeile angegeben werden) und seinem Paßwort (falls benötigt) gefragt.

Falls die Anmeldung erfolgreich war, aktualisiert **login** Protokolldateien, setzt verschiedene Umgebungsvariablen, teilt dem Benutzer mit, ob er Post erhalten hat, und führt die Startdateien aus.

Niemand mit Ausnahme von **root** kann sich anmelden, wenn */etc/nologin* existiert. Diese Datei wird vor Abbruch einer Verbindung angezeigt. **root** kann sich in diesem Fall ausschließlich von einem TTY-Terminal anmelden, das in */etc/securetty* benannt ist. Existiert *~/.hushlogin*, erfolgt die Anmeldung ohne Ausgabe von Statusmeldungen. Ist */var/adm/lastlog* vorhanden, wird der Zeitpunkt der letzten Anmeldung ausgegeben.

Optionen

-f
: Soll die Authentifizierung einer zweiten Login-Prozedur umgehen, wird aber unter Linux nicht unterstützt.

-h *Hostname*
: Dient beim Login von einem entfernten Rechner zur Übergabe des *Hostnamens* vom Ursprungsrechner der Verbindung. Diese Option ist nur verfügbar, wenn der Login-Prozeß mit Superuser-Rechner läuft.

-p
: Unterdrückt das Setzen von Umgebungsvariablen. Die vorherige Konfiguration wird beibehalten.

logname

logname

logname durchsucht */etc/utmp* nach dem Loginnamen des Benutzers. Bei Erfolg wird er ausgegeben, ansonsten wird eine Fehlermeldung angezeigt.

look [*Optionen*] *Text* [*Datei*]

look

look sucht in *Datei* (Vorgabe: */usr/dict/words*) nach Textzeilen, die mit *Text* anfangen.

Optionen

-d
: Vergleicht nur alphanumerische Zeichen.

-f
: Sucht ohne Berücksichtigung der Groß-/Kleinschreibung.

-t *Zeichen*
: Untersucht nur Zeichen bis einschließlich *Zeichen*.

lpq [*Optionen*] [*Name*]

lpq

lpq überprüft die Druckerwarteschlange nach dem Status von Druckaufträgen. Für jeden Auftrag werden Benutzername, Position in der Warteschlange, Dateiname, Auftragsnummer und Dateigröße in Bytes angezeigt. Ist *Name* angegeben, so werden nur die Informationen über Aufträge des angegebenen Benutzers angezeigt.

→

lpq
←

Optionen

-P*Drucker*
: Bestimmt, für welchen Drucker die Warteschlange überprüft werden soll. Fehlt diese Angabe, so wird der Systemdrucker oder der in der PRINTER- Umgebungsvariablen angegebene Drucker angenommen.

-l
: Druckt Informationen über jede Datei, die von einem Druckauftrag betroffen ist.

#*num*
: Prüft den Status einer bestimmten Auftragsnummer.

lpr

lpr [*Optionen*] *Dateien*

lpr fügt *Dateien* der Druckerwarteschlange hinzu.

Optionen

-c
: Erwartet Daten, die von **cifplot** stammen.

-d
: Erwartet Daten im DVI- (Device-independent-) Format, die von TEX stammen.

-f
: Benutzt einen Filter, der das erste Zeichen einer Zeile als ein Standard-'carriage-control'-Zeichen interpretiert.

-g
: Erwartet plot-Daten, die dem Standard der **plot**-Routinen entsprechen.

-l
: Benutzt einen Filter, der den Druck von Steuerzeichen erlaubt und Seitenvorschübe unterdrückt.

-n
: Erwartet Daten von **ditroff** (device-independent **troff**).

-p
: Benutzt **pr** zum Formatieren der Dateien (entspricht **print**).

-t
: Erwartet Daten von **troff** (Cat-phototypesetter-Kommandos).

-v
: Erwartet Rasterdaten für Geräte wie den Benson Varian.

-P*Drucker*
: Fügt die Dateien der Warteschlange für *Drucker* hinzu.

-h
: Unterdrückt den Ausdruck der 'Burst'-Seite

-m
: Schickt eine Nachricht an den Benutzer, wenn der Druckauftrag ausgeführt wurde.

-r
: Entfernt die Dateien nach erfolgtem Ausdruck (zusammen mit der -s- Option).

-s
: Benutzt symbolische Verknüpfungen, anstatt die Dateien in das Spool-Verzeichnis zu kopieren. Dies spart Zeit und Plattenplatz bei großen Druckaufträgen. Die Dateien dürfen vor Abschluß des Druckvorganges weder verändert noch entfernt werden (automatisches Entfernen mit -s).

-#*Anzahl*
: Druckt *Anzahl* Kopien der angegebenen Dateien.

-C *Text*
: Ersetzt den Systemnamen auf der 'Burst'-Seite durch *Text.*

-J *Text*
: Ersetzt den Dateinamen auf der 'Burst'-Seite durch *Text.*

-T *Titel*
: Benutzt *Titel* als Titelbezeichnung, wenn **pr** verwendet wird.

-i [*Spalten*]
: Rückt die Ausgabe ein. Der Vorgabewert ist 8 und kann durch Angabe des *Spalten*-Arguments geändert werden.

-w *Anzahl*
: Setzt *Anzahl* Zeichen als zu verwendende Seitenbreite für **pr**.

lpr

lprm [*Optionen*] [*Auftragsnummer*] [*Benutzer*]

lprm

lprm entfernt einen Druckauftrag aus der Druckerwarteschlange. Es wird die Angabe der Auftragsnummer benötigt, wie sie von **lpq** geliefert wird. Ein privilegierter Benutzer kann statt dessen auch einen *Benutzer* angeben, dessen sämtliche Druckaufträge gelöscht werden sollen.

Optionen

-P*Drucker*
: Bestimmt, bei welchem Drucker Aufträge aus der Warteschlange entfernt werden sollen. Fehlt diese Angabe, so wird der Systemdrucker oder der in der PRINTER- Umgebungsvariablen angegebene Drucker angenommen.

- Entfernt alle Aufträge in der Warteschlange, die zu einem bestimmten *Benutzer* gehören.

ls [*Optionen*] [*Dateien*]

ls

ls zeigt den Inhalt von Verzeichnissen. Sind keine *Dateien* angegeben, werden die Dateien im aktuellen Verzeichnis aufgelistet. Sind eine oder mehrere *Dateien* angegeben, werden entweder die Dateien im Verzeichnis *Name* aufgelistet oder alle Dateien, deren Name zu *Name* paßt. Dabei darf *Dateien* Sonderzeichen enthalten. Die verschiedenen Optionen erlauben die Anzeige einer Vielzahl von Informationen in unterschiedlichen Formaten. Die nützlichsten Optionen sind unter anderem **-F**, **-R**, **-l** und **-s**. Einige Optionen hingegen schließen sich gegenseitig aus, wie **-u** und **-c**.

Optionen

-1, **--format=single-column**
: Gibt jeden Dateieintrag in einer eigenen Zeile aus

-a
: Listet alle Dateien auf, auch die normalerweise versteckten Dateien, deren Namen mit einem Punkt beginnen.

-b, **--escape**
: Zeigt nicht darstellbare Zeichen als Oktalzahl an.

-c, **--time-ctime**, **--time=status**
: Sortiert die Dateien anhand der Zeit ihrer letzten Statusänderung (nicht anhand der letzten Inhaltsveränderung).

-d, **--directory**
: Listet nur die Verzeichnisnamen auf, nicht ihren Inhalt.

→

-f Unterdrückt das Sortieren der Dateinamen. Die Ausgabe erfolgt entsprechend der physikalischen Position im Verzeichnis.

-i, --inode
Zeigt die Inode-Nummer für jede Datei an.

-k, --kilobytes
Gibt Dateilängen in Kilobytes aus. Die Umgebungsvariable POSIXLY_CORRECT wird ignoriert.

-l, --format=long, --format=verbose
Gibt Daten im erweiterten Format aus. Dies beinhaltet Zugriffsrechte, Besitzer, Länge, Zeitpunkt der letzten Änderung und mehr.

-m, --format=commas
Gibt eine durch Kommata getrennte Namensliste aus.

-n, --numeric-uid-gid
Entspricht -l, jedoch werden statt des Besitzernamens und der Gruppe die GruppenID- und BenutzerID-Nummer ausgegeben.

-p Markiert Verzeichnisse, indem ein / an ihren Namen angehängt wird.

-q, --hide-control-chars
Zeigt nicht darstellbare Zeichen als ? an.

-r, --reverse
Dreht die Sortierreihenfolge um.

-s, --size
Zeigt die Dateilänge in Blöcken zu 512 Bytes an, wenn POSIXLY_CORRECT gesetzt ist. Ansonsten wird die Länge in Kilobytes ausgegeben.

-t, --sort=time
Sortiert anhand des Zeitpunktes der letzten Änderung (neueste zuerst).

-u, --time=atime, --time=access, --time=use
Sortiert anhand des Zeitpunktes des letzten Zugriffes.

-x, --format=across, --format=horizontal
Gibt die Namen horizontal sortiert aus.

-A, --almost-all
Entspricht *-l*, jedoch werden . und .. nicht mit angezeigt.

-B, --ignore-backups
Ignoriert Dateien, die auf ~ enden, sofern sie nicht explizit angefordert werden.

-C, --format=vertical
Gibt die Namen in vertikal sortierten Spalten aus (Voreinstellung).

-F, --classify
Klassifiziert Dateieinträge durch Anhängen folgender Zeichen: / für Verzeichnisse, * für Programmdateien, @ für symbolische Verknüpfungen, | für FIFOs und = für Sockets.

-G, --no-group
Unterdrückt den Gruppennamen im erweiterten Format.

-L, --dereference
: Listet statt einer symbolischen Verknüpfung die Datei oder das Verzeichnis, auf die oder das diese Verknüpfung zeigt.

-N, --literal
: Gibt die Namen aus, ohne sie in " einzufassen.

-Q, --quote-name
: Faßt Dateinamen in " ein. Nicht druckbare Zeichen werden als Oktalzahl dargestellt.

-R, --recursive
: Gibt rekursiv den Inhalt von Unterverzeichnissen aus.

-S, --sort=size
: Sortiert nach Größe, beginnend mit den größten Dateien.

-U, --sort=none
: Unterdrückt das Sortieren der Dateinamen entsprechend -f, benutzt jedoch zur Ausgabe das erweiterte Format.

-X, --sort=extension
: Sortiert anhand der Dateinamenserweiterung.

Beispiele

Alle Dateien des aktuellen Verzeichnisses sollen mit ihrer Länge aufgelistet werden. Dabei sollen besondere Dateien markiert und die Namen in Spalten ausgegeben werden:

```
ls -asCF
```

Der Status der Verzeichnisse */bin* und */etc* soll ausgegeben werden:

```
ls -ld /bin /etc
```

Alle C-Quelltexte im aktuellen Verzeichnis sollen beginnend mit der ältesten Datei aufgelistet werden:

```
ls -rt *.c
```

Die Anzahl der (sichtbaren) Dateien im aktuellen Verzeichnis soll ermittelt werden:

```
ls | wc -l
```

lsattr [*Optionen*] [*Dateien*]

lsattr

lsattr gibt die Attribute von Dateien auf einem „Linux Second Extended File System“ aus. Siehe auch **chattr**.

Optionen

-a
: Zeigt alle (auch „unsichtbare“) Dateien in den angegebenen Verzeichnissen an.

-d
: Zeigt die Verzeichnisattribute anstatt der Attribute der enthaltenen Dateien an.

-R
: Zeigt rekursiv auch die Inhalte von Unterverzeichnissen an.

-v
: Gibt die Version der Dateien aus.

mail

mail [*Optionen*] [*Empfänger*]

Post lesen oder an andere *Empfänger* senden. Das **mail**-Programm ermöglicht das Erstellen, Senden, Empfangen, Weiterleiten und Beantworten von elektronischer Post. **mail** hat zwei Hauptmodi: den Schreibmodus, in dem eine neue Nachricht erzeugt werden kann, und den Kommandomodus, in dem die Mailbox verwaltet wird.

mail ist ein mächtiges Werkzeug, aber es ist für den neuen Anwender etwas schwierig zu bedienen. Die meisten Linux-Distributionen enthalten daher **pine** und **elm**, die ähnliche Funktionen mit etwas mehr Komfort anbieten.

Dieser Abschnitt stellt **mail**- Befehle, -Optionen und -Dateien vor. Um Ihnen einen schnellen Einstieg zu ermöglichen, hier kurz die beiden wichtigsten Kommandos:

In den interaktiven Modus zum Lesen der Post gelangt man mit dem Kommando:

```
mail
```

Um eine Nachricht an einen bestimmten *Empfänger* zu schreiben, ruft man das Programm so auf:

`mail` *`Empfänger`*

Kommandozeilenoptionen

-b *Liste*
: Verändert das verborgene „Weitere Empfänger"-Feld in eine durch Kommata getrennte *Liste*.

-c *Liste*
: Verändert das „Weitere Empfänger"-Feld in eine durch Kommata getrennte *Liste*.

-f [*Datei*]
: Verarbeitet den Inhalt der *Datei* oder mbox anstatt */var/spool/mail/$USER*.

-i
: Keine Antwort auf tty-Unterbrechungssignale.

-n
: Ingnoriert */etc/mail.rc* bei dem Start.

-p
: Liest Post im POP-Modus.

-s *Betreff*
: Ändert Betreff in *Betreff*.

-u
: Verarbeitet den Inhalt */var/spool/mail/$USER*. Standard.

-v
: Alle Meldungen. Gibt die Informationen über den Versand der Post auf die Standardausgabe aus.

-I
: Interaktiv, auch wenn die Standardeingabe auf ein anderes Terminal umgelenkt ist.

-N
: Beim Drucken einer Mail oder beim Wechseln in ein Mail-Verzeichnis keinen Nachrichten-Header anzeigen.

-P
: Schaltet den POP-Modus aus.

Befehle im Schreibmodus

~! Aus dem Schreibmodus in eine Shell wechseln.

~? Gibt eine Liste der Sonderzeichen im Schreibmodus aus.

~b Fügt einen *Bcc:* Header hinzu oder verändert ihn.

~c Fügt einen *Cc:* Header hinzu oder verändert ihn.

~d Liest die *dead.letter*-Datei ein.

~e Ruft den Editor auf.

~f*Nachrichten*
: Fügt *Nachrichten* in die gerade erzeugte Nachricht ein.

~F*Nachrichten*
: Vergleichbar mit ~f, aber mit Nachrichten-Header.

~h Fügt alle Nachrichten-Header hinzu oder verändert sie.

~m*Nachrichten*
: Vergleichbar mit ~f, aber mit einem Tab eingerückt.

~M*Nachrichten*
: Vergleichbar mit ~m, aber mit Nachrichten-Header.

~p Zeigt Nachrichten-Header-Felder und die versandte Nachricht an.

~q Bricht das Schreiben der aktuellen Nachricht ab.

~r Hängt Datei an die aktuelle Nachricht an.

~s Hängt einen *Betreff:*-Header an oder verändert ihn.

~t Fügt Namen in die *To:*-Liste ein.

~v Ruft den in der VISUAL-Umgebungsvariablen stehenden Editor auf.

~| *Kommando*
: Leitet die Nachricht durch ein *Kommando*.

~:*mail-Kommando*
: Führt das *mail-Kommando* aus.

~*Zeichenkette*
: Fügt *Zeichenkette* ein. Wenn die Zeichenkette ~ enthält, muß sie mit einem \ entwertet werden.

Befehle im Kommandomodus

?	Zeigt eine Zusammenfassung der Befehle (Hilfeseite).
!	Führt ein Shell-Kommando aus.
- *Anzahl*	Gibt die um *Anzahl* zurückliegende Nachricht aus; Standard ist die unmittelbar zurückliegende.
alias (a)	Gibt die Alias-Liste aus oder erzeugt sie.
alternates (alt)	Bezeichnet die Remote Accounts auf Remote-Maschinen. Veranlaßt mail, keine Antworten dorthin zu senden.
chdir (c)	cd in das Heimatverzeichnis oder das angegebene Verzeichnis.
copy (co)	Vergleichbar mit save (sichern), markiert Nachrichten, aber nicht zum Löschen.
delete (d)	Löscht eine Nachricht.

→

mail ←

dp	Löscht die aktuelle Nachricht und zeigt die nächste an.
edit (e)	Editiert eine Nachricht.
exit (ex, x)	Beendet mail, ohne den Ordner zu aktualisieren.
file (fi)	Wechselt den Ordner.
folder (fold)	Liest die in einer Datei gespeicherten Nachrichten ein. Mögliche Dateien sind: # vorhergehend % Systemmailbox %*Benutzer* Die Systemmailbox des *Benutzers* & mbox
+*Ordner*	Datei im *Ordner*-Verzeichnis.
folders	Zeigt die Ordner an.
headers (h)	Zeigt die Nachrichten-Header auf der aktuellen Eingabeaufforderung an.
headers+ (h+)	Bewegt ein Fenster mit Headern vorwärts.
headers- (h-)	Bewegt ein Fenster mit Headern rückwärts.
help	Vergleichbar mit ?.
hold (ho)	Hält die Nachrichten in dem System-Briefkasten.
ignore	Fügt der Liste der zu ignorierenden Felder weitere Felder hinzu.
mail *Empfänger* (m)	Schreibt eine Nachricht an *Empfänger*.
mbox	Standard. Bewegt die bezeichneten Nachrichten beim Beenden nach mbox .
next (n)	Zeigt die nächste Nachricht oder die nächste Nachricht, auf die das Argument zutrifft.
preserve (pr)	Ein Synonym für **hold**.
print [*Liste*] (p)	Zeigt jede Nachricht an, die in der *Liste* vorkommt.
Print [*Liste*] (P)	Vergleichbar mit **print**, zeigt jedoch die Header-Felder mit an.
quit (q)	Beendet mail und aktualisiert den Ordner.
reply (r)	Sendet eine Mail an alle Mitglieder der Verteilungsliste.
Reply (R)	Antwortet ausschließlich dem Autor der Mail.
respond	Vergleichbar mit **reply**.
retain	Bindet immer die in der Liste enthaltenen Header-Felder mit ein wenn Nachrichten angezeigt werden. Ohne Argumente werden die zurückgehaltenen Felder angezeigt.
save (s)	Sichert Nachrichten in einem Ordner.
saveignore	Entfernt die zu ignorierenden Felder beim Speichern.
saveretain	Übergeht **saveignore**, um die angegebenen Felder zu behalten.
set (se)	Setzt die **mail**-Optionen oder gibt den aktuellen Status aus.
shell (sh)	Eröffnet eine neue Shell.
size	Gibt die Größe der angegebenen Nachrichten aus.
source	Liest Kommandos aus einer Datei.

top	Zeigt die ersten paar Zeilen der angegebenen Nachrichten an.
type (t)	Vergleichbar mit **print**.
Type (T)	Vergleichbar mit **Print**.
unalias	Verwirft die vorher definierten Aliase.
undelete (u)	Stellt gelöschte Nachrichten wieder her.
unread (U)	Markiert die angegebenen Nachrichten als ungelesen.
unset (uns)	Setzt die **mail**-Optionen zurück.
visual (v)	Editiert die Nachrichten mit dem in der VISUAL-Variablen angegebenen Editor.
write (w)	Schreibt eine Nachricht ohne Header in eine Datei.
xit (x)	Vergleichbar mit **exit**.
z	Lenkt die Aufmerksamkeit von **mail** auf die nächste Bildschirmseite mit Text. Mit **z-** kehrt man zurück.

mail Optionen

Diese Optionen werden innerhalb der *.mailrc*-Datei verwendet. Die Syntax ist „set *Option*" oder „unset *Option*".

append	Fügt Nachrichten mbox hinzu.
ask	Erwartet Eingabe des Betreffs.
askcc	Erwartet die Eingabe für weitere Empfänger.
asksub	Erwartet Eingabe für **Subject:** (Betreff) Zeile.
chron	Zeigt die Nachrichten in zeitlicher Abfolge, die neuesten zuletzt.
debug	Vergleichbar mit **-d** auf der Kommandozeile.
dot	Verwendet einen einzelnen . als *EOF* (Dateiende).
folder	Bestimmt ein Verzeichnis, in dem die Nachrichten-Ordner liegen.
hold	Behält beim Beenden die Nachrichten in der System-Mailbox .
ignore	Ignoriert Interrupt-Signale vom Terminal. Gibt sie als @ aus.
ignoreeof	Behandelt **^D** nicht als ein *EOF*.
metoo	Entfernt Absender nicht aus Gruppen, wenn sie angeschrieben werden.
noheader	Vergleichbar mit **-N** auf der Kommandozeile.
nokerberos	Empfängt POP-Post via POP3-, nicht mit dem KPOP-Protokoll.
nosave	Speichert abgebrochene Briefe nicht in *dead.letter*.
pop-mail	Empfängt Post mit dem POP3-Protokoll und speichere sie in *mbox.pop*.
prompt	Ändert den Text der Eingabeaufforderung.
Replyall	Ändert die Rollen von **Reply** und **reply**.
quiet	Gibt keine Versionsnummer beim Start aus.
searchheaders	Bei der Angabe der Bezeichner */x:y* erweitert alle Nachrichten, welche die Zeichenkette *y* im *x*-Header enthalten.
verbose	Vergleichbar mit **-v** auf der Kommandozeile.

→

mail ←

verbose-pop — Zeigt den aktuellen Status während des Empfangs von POP-Post an.

Spezielle Dateien

calendar	Enthält Erinnerungen, welche das Betriebssystem an Sie schickt.
.maildelivery	Mail-Transport-Konfigurationsdatei.
.mailrc	Mail-Konfigurationsdatei.
triplog	Speichert die Empfänger Ihrer automatischen Antworten.
tripnote	Enthält die automatischen Antworten.

---

man

man [*Optionen*] [*Abschnitt*] [*Titel*]

Zeigt die Online-Handbücher an. **man** findet und gibt die benannten *Titel* und *Abschnitte* aus.

Optionen

-7, --ascii
: Erwartet eine reine ASCII-Datei und formatiert sie für ein 7-Bit-Terminal oder einen Terminal-Emulator.

-a, --all
: Zeigt alle Seiten, auf die *title* zutrifft.

-b
: Zeigt die Datei mit Leerzeichen an.

-d, --debug
: Zeigt Informationen zur Fehlersuche an. Unterdrückt die aktuelle Ausgabe des Handbuchs.

-f, --whatis
: Wie das **whatis**-Kommando.

-k, --apropos
: Wie das **apropos**-Kommando.

-l, --local-file
: Durchsucht lokale Dateien, keine Systemfiles nach Online-Handbüchern. Wenn **i** als *Dateiname* angeben wird, wird die Standardeingabe durchsucht.

-m *System*, **--systems=***System*
: Durchsucht *System*-Handbuch-Seiten. *System* muß eine durch Kommata getrennte Liste sein

-p *Präprozessoren*, **--Präprozessor=***Präprozessoren*
: Bearbeitet Online-Handbücher mit *Präprozessoren*, bevor sie an **nroff**, **troff** oder **groff** übergeben werden. Startet immer zuerst **soelim**.

-r *Eingabeaufforderung*, **--prompt=***Eingabeaufforderung*
: Setzt die Eingabeaufforderung, wenn **less** als Anzeigeprogramm verwendet wird.

-t, --troff
: Formatiert die Handbuchseite mit */usr/bin/groff -Tgv -mandoc*. Indirekt mit -T und -Z.

-u, --update
: Überprüft die Übereinstimmung zwischen Handbuch-Cache und Dateisystem.

-w, --where, --location
: Gibt den Pfadnamen des Handbuches an, ohne es anzuzeigen.

-D, --default
: Setzt alle Optionen auf Standardwerte zurück.

-L *locale*, --locale=*locale*
: Vermutet aktuellen Ort als *locale*, ohne **setlocale()** zu beachten.

-M *Pfad*, --manpath=*Pfad*
: Sucht nach Online-Handbüchern im *path*. Ignoriert die -m-Option.

-P*Anzeigeprogramm*, —pager=*Anzeigeprogramm*
: Wählt ein Anzeigeprogramm *Anzeigeprogramm* für die Ausgabe.

-T *Gerät*, --troff-device[=*Gerät*]
: Formatiert **groff**- oder **troff**-Ausgabe für *Gerät*, wie z.B. **dvi**, **latin1**, **X75** und **X100**.

-Z, --ditroff
: Erlaubt keine Nachbearbeitung von Online-Handbüchern, nachdem **groff** seine Arbeit beendet hat.

Abschnitsnamen

Online-Handbücher sind in Abschnitte aufgeteilt.

1	Ausführbare Programme und Shell-Kommandos
2	System-Aufrufe (Funktionen, die vom Kernel bereitgestellt werden)
3	Bibliotheken-Aufrufe (Funktionen innerhalb System-Bibliotheken)
4	Spezielle Dateien (gewöhnlich in */dev* zu finden)
5	Datei-Formate und Konventionen, z.B. */etc/passwd*
6	Spiele
7	Makro-Pakete und Konventionen
8	System-Administrator-Kommandos (gewöhnlich nur für den Superuser root)
9	Kernel-Routinen (Kein Standard)

manpath [*Optionen*]

Versucht, den Pfad zu den Online-Handbüchern zu bestimmen. Überprüft zuerst, ob die Variable $MANPATH gesetzt ist; wenn sie nicht gesetzt ist, suche in */etc/manpath.conf*, in Anwender-Umgebungsvariablen und im aktuellen Arbeitsverzeichnis.

Optionen

-q, --quiet
: Gibt keine Warnungen aus.

-d, --debug
: Gibt Debugging- (Fehlersuche-) Informationen aus .

→

manpath
←

-c, --**catpath**
: Wandelt den ermittelten **manpath** in den relativen **catpath** um.

-g, --**global**
: Erstellt **manpath** aus der Liste von Pfaden, die in */etc/manpath.conf* als global bezeichnet sind.

-m *System*, --**systems**=*System*
: Sucht auf *Systemen* nach dem Pfad.

mesg

mesg [*Option*]

Ändert die Fähigkeit anderer Anwender, **write**-Nachrichten an Ihr Terminal zu senden. Ohne Optionen wird der aktuelle Status angezeigt.

Optionen

n
: Verbietet **write**-Nachrichten.

y
: Erlaubt **write**-Nachrichten (Standard).

mkdir

mkdir [*Optionen*] *Verzeichnisse*

Erstellt ein oder mehrere *Verzeichnisse*. Um ein Verzeichnis erstellen zu können, muß man Schreibberechtigung im betreffenden Verzeichnis haben. Siehe auch **rmdir**. Die Standardzugriffsberechtigung von neuen Verzeichnissen ist 0777, verändert durch das System oder des Anwenders **umask**.

Optionen

-m, --**mode** *Modus*
: Setzt den Zugriffsberechtigungs-*Modus* für neue Verzeichnisse. Siehe **chmod** für eine Erläuterung der verschiedenen Zustände von *Modus*.

-p, --**parents**
: Legt Unterverzeichnisse in einem nicht existierenden Verzeichnis an.

Beispiele

Erstellt ein schreibgeschütztes Verzeichnis **Personal**:

```
mkdir -m 444 Personal
```

Die folgenden Eingaben:

```
mkdir Arbeit; cd Arbeit
mkdir nil; cd nil
mkdir Fragen; cd ../..
```

können vereinfacht werden durch:

```
mkdir -p Arbeit/nil/Fragen
```

more [*Optionen*] [*Dateien*]

Zeigt die angegebenen *Dateien* Seite für Seite an. Siehe **less** als eine Alternative zu **more**. Manchen Kommandos kann eine Nummer vorangestellt werden.

Optionen

+*Zahl* Beginnt Anzeige ab Zeile *Zahl*.

+/*Muster*
: Beginnt mit der Anzeige zwei Zeilen vor *Muster*.

-c Erneuert die Anzeige von oben her, anstatt zu blättern.

-d Anzeige des Hinweises „Zum Fortfahren Leertaste drücken, q zum Abbrechen", als Antwort auf unzulässige Eingaben; schaltet den Warnton ab.

-f Zählt die Textzeilen anstatt der Bildschirmzeilen. Nützlich, wenn lange Zeilen außerhalb des Anzeigebereichs umbrochen werden. Ignoriert (**CTRL-L**) Seitenvorschub. Bedingt, daß ein Wagenrücklauf als **CTRL-M** eingegeben werden muß.

-num *Anzahl*
: Setzt die Darstellung auf *Anzahl* Zeilen.

-p Stellt jede Seite neu dar, anstatt zu rollen. Das ist manchmal schneller.

-r Erzwingt die Anzeige von Steuerzeichen in der Form ^*x*.

-s Zusammenpressen; zeigt mehrere Leerzeichen als eins an.

-u Unterdrückt unterstrichene Buchstaben.

Befehle

Alle Befehle in **more** basieren auf **vi**-Befehlen. Vor vielen Befehlen kann ein Argument angegeben werden.

Leertaste Zeigt die nächste Seite des Textes an.

z Zeigt die nächsten *Zeilen* des Textes an und definiert eine Bildschirmseite zu *Zeilen* Zeilen um. Standard ist eine Bildschirmseite.

ENTER Zeigt die nächsten *Zeilen* des Textes, definiert eine Bildschirmseite zu *Zeilen* Zeilen um. Default ist eine Zeile.

d, ^D Rollt *Textzeilen* und definiert die Anzahl der zu rollenden *Zeilen* um zu Zeilen. Standard ist eine Zeile.

q, Q, Abbruch
: Beenden.

s Überspringt eine Textzeile.

f Überspringt eine Bildschirmseite.

b, ^B Springt eine Bildschirmseite zurück.

' Kehrt an den Ort der ersten Suche zurück.

= Gibt die aktuelle Zeilennummer aus.

/*Muster* Sucht nach *Muster*, springt nach *Nummer* Vorkommen, wenn ein Argument angegeben wurde.

n Wiederholt letzte Suche, springt zu *Nummer* Vorkommen, wenn ein Argument angegeben wurde.

→

more
←

!*Kommando*, :!*Kommando*
: Ruft eine Shell auf und führt das *Kommando* dort aus.

v
: Ruft den Editor **vi** mit der aktuellen Datei an der aktuellen Zeile auf.

^L
: Zeichnet den Bildschirm neu.

:n
: Springt zur nächsten Datei.

:p
: Springt zur vorherigen Datei.

:f
: Gibt aktuellen Dateinamen und Zeilennummer aus.

.
: Führt voriges Kommando erneut aus.

Beispiele

Blättert durch *Datei* im „freien" Modus mit Eingabeaufforderungen:

```
more -cd Datei
```

Formatiert *doc* auf dem Bildschirm und entfernt Unterstreichungen:

```
nroff doc | more -u
```

Zeigt die Dokumentation des **grep**-Kommandos an; beginnt nahe dem Wort „BUGS" und faßt zusätzliche Leerzeichen zusammen:

```
man grep | more +/BUGS -s
```

mv

mv [*Option*] *Quelle Ziel*

Verschiebt oder benennt Dateien und Verzeichnisse um. Die Quelle (erste Spalte) und das Ziel (zweite Spalte) bestimmen folgendes Ergebnis (dritte Spalte):

Quelle	*Ziel*	*Ergebnis*
Datei	*Name* (nicht vorhanden)	Benennt Datei um in *Name*.
Datei	Vorhandene Datei	Überschreibt existierende Datei mit der Quelldatei.
Verzeichnis	*Name* (nicht vorhanden)	Benennt Verzeichnisse um in *Name*.
Verzeichnis	Vorhandenes Verzeichnis	Verschobenes Verzeichnis wird ein Unterverzeichnis im existierenden Verzeichnis.
Eine oder mehrere Dateien	Vorhandenes Verzeichnis	Verschiebt die Dateien in ein Verzeichnis.

Optionen

-b, --backup
: Erzeugt eine Sicherungskopie, bevor die Dateien gelöscht werden.

-f
: Verschiebt die Quelle, auch wenn das *Ziel* vorhanden ist; unterdrückt Meldungen über fehlende Zugriffsrechte.

-i, --interactive
: Verlangt eine Bestätigung, bevor Dateien gelöscht werden.

-u, --update
: Verschiebt Dateien nur, wenn sie neuer sind als die gleichnamigen Dateien.

newgrp [*Gruppe*]

Vergleichbar zu **login**. Ändert die aktive Gruppenkennung des Anwenders. Ohne Angabe einer Gruppe wird zu der in der */etc/passwd* festgelegten Standardgruppe des Anwenders gewechselt. Sonst wird in die angegebene Gruppe gewechselt, wenn der Anwender in der Datei */etc/group* als Mitglied dieser Gruppe eingetragen ist.

nice [*Option*] *Kommando* [*Argumente*]

Führt ein *Kommando* und *Argumente* mit einer veränderten Priorität aus (i.e. „nett" sein zu anderen Anwendern). Ohne Argumente wird die standardmäßige Priorität angezeigt (Grad der Nettigkeit). Wenn **nice** ein „Kind"-Prozeß ist, wird die Priorität des Elternprozesses ausgegeben. Die Nettigkeit kann in einem Bereich von –20 (höchste Priorität) bis zu 19 (niedrigste Priorität) liegen.

Option

-*adjustment*, --**adjustment=***adjustment*
: Startet *Kommando* mit einer Nettigkeit erhöht um *adjustment* (1–19); Standard ist 10. Ein privilegierter Anwender **root** kann die Priorität erhöhen, indem er ein negatives *adjustment* verwendet (z.B. –5).

nohup *Kommando* [*Argumente*]

Führt weiterhin das bezeichnete *Kommando* und wahlweise Programm *Argumente* aus, nachdem sich der Anwender ausgeloggt hat. (Schützt Programme vor den Signalen SIGHUP, SIGINT, SIGQUIT und SIGTERM; i.e. **kein hangup**). TTY-Ausgaben werden standardmäßig an die Datei *nohup.out* angehängt. Moderne Shells erhalten Hintergrundprozesse standardmäßig; dieses Kommando ist nur bei der originalen Bourne Shell von Bedeutung.

passwd [*Anwender*]

Erzeugt oder verändert das Paßwort des *Anwenders*. Nur der Anwender selbst oder der Superuser root kann ein Paßwort ändern. Normale Benutzer müssen zum Ändern ihres Paßwortes ihren Namen nicht angeben.

paste [*Optionen*] *Dateien*

Fügt die Zeilen von zwei oder mehr *Dateien*, getrennt durch einen Tabulator, horizontal zusammen. Siehe auch **cut**, **join**, **newform** und **pr**.

→

paste ←

Optionen

- Ersetzt einen Dateinamen mit der Standardeingabe.

-d'*Zeichen*'
: Trennt die Spalten mit *Zeichen* anstelle eines Tabulators. Anmerkung: Spalten können mit unterschiedlichen Zeichen getrennt werden, wenn mehr als ein *Zeichen* angegeben wird.

-s, --serial
: Fügt fortlaufende Zeilen zu einer Datei zusammen.

Beispiele

Erzeugt eine drei-spaltige *Datei* aus den Dateien *x*, *y* und *z*:

```
paste x y z > Datei
```

Erzeugt eine zweispaltige Liste der aktuellen Benutzer:

```
who | paste - - > Datei
```

Erzeugt aus allen Zeilen einer Datei eine einzelne Zeile:

```
paste -s -d"\t\n" list
```

pathchk

pathchk [*Optionen*] *Dateinamen*

Überprüft die Gültigkeit und Übertragbarkeit von *Dateinamen*. Speziell wird überprüft, ob alle Verzeichnisse im Pfad durchsuchbar sind und ob die Länge der Dateinamen korrekt ist.

Option

-p, --portability
: Überprüft die Übertragbarkeit auf alle POSIX-kompatiblen Systeme.

perl

perl

perl ist eine mächtige Programmiersprache zur Textverarbeitung, die viele der nützlichsten Methoden von Shell, C, **awk** und **sed** kombiniert und durch eigene Features erweitert. Weitere Informationen finden Sie in Kapitel 11.

pidof

pidof [*Optionen*] *Programme*

Zeigt die Prozeßidentifikationsnummern der in der Kommandozeile angegebenen Programme an. **pidof** ist ein symbolischer Link auf **killall5**.

Optionen

-s
: Liefert eine einzelne Prozeßidentifikationsnummer.

-o *PIDs*
: Überspringt die Prozesse mit den angegebenen Identifikationsnummern *PIDs*. Sie können eine Liste mit mehreren Nummern angeben.

popclient [*Optionen*] *Host*

popclient

Holt Mail unter Verwendung des „Post Office Protocols". Das Programm unterstützt POP2 und POP3.

Optionen

-2 Erzwingt die Benutzung von POP2.

-3 Erzwingt die Benutzung von POP3.

-c Veranlaßt die Ausgabe der empfangenen Nachrichten auf die Standardausgabe.

-f *Maildatei*
Veranlaßt die Übertragung der angegebenen *Maildatei* vom entfernten Server.

-k Veranlaßt den Server, die Maildatei nach der Abholung nicht zu löschen.

-o *Maildatei*
Veranlaßt das Programm, die empfangenen Nachrichten an die angegebene *Maildatei* anzuhängen. Die voreingestellte Maildatei ist in der Regel *var/spool/mail/Benutzer*.

-p *Paßwort*
Veranlaßt die Verwendung des angegebenen *Paßworts* zum Einloggen in das entfernte System. Ohne dieses Argument wird das Paßwort interaktiv erfragt.

-s Unterdrückt die Ausgabe von Meldungen und Fortschrittsanzeigen während der Verbindung mit dem entfernten Rechner.

-u *Benutzername*
Veranlaßt die Benutzung des *Benutzernamens* zum Einloggen in das entfernte System. Voreinstellung ist der lokale Name des Benutzers.

-v Veranlaßt die Ausgabe aller Kontrollmeldungen auf die Standardfehlerausgabe. Diese Option verdrängt den Schalter -s.

pr [*Datei*]

pr

Konvertiert eine Textdatei in eine eventuell mehrspaltig formatierte Version mit Seitennummern und Kopfzeilen. Wenn anstelle des Dateinamens ein - angegeben ist, liest das Programm von der Standardeingabe.

Optionen

+*Zahl* Überspringt die ersten *Zahl*-1 Seiten und beginnt mit der Ausgabe der Seite *Zahl*.

-*Anzahl*
Veranlaßt die Formatierung in der angegebenen *Anzahl* Spalten.

-a Veranlaßt das Auffüllen horizontaler Spalten vor den vertikalen Spalten.

-b Veranlaßt den Ausgleich horizontaler Spalten am Dateiende.

-c Veranlaßt die Umwandlung von Kontrollzeichen in Caret-Notation (wie z.B. ^C) und anderer nichtdruckbarer Zeichen in Oktalzahldarstellung mit Backslash.

→

pr
←

-d Veranlaßt die Ausgabe mit doppeltem Zeilenabstand.

-e[*Tab-Zeichen*[*Weite*]]
: Verwandelt Tabulatoren (oder die angegebenen *Tab-Zeichen*) in Folgen von Leerzeichen. Wenn eine *Weite* angegeben ist, werden die Tabulatoren in die entsprechende Anzahl Zeichen umgewandelt (Voreinstellung ist 8).

-f, -F Veranlaßt die Trennung der Seiten durch Seitenvorschub anstelle von Zeilenvorschub.

-h *Kopfzeile*
: Veranlaßt die Benutzung der angegebenen *Kopfzeile* anstelle des Dateinamens.

-l *Anzahl*
: Setzt die Seitenlänge auf die angegebene *Anzahl* Zeilen (Voreinstellung ist 66). Wenn die Zeilenzahl kleiner ist als 10, wird die Ausgabe von Kopf- und Fußzeilen unterdrückt.

-m Veranlaßt die parallele Ausgabe von jeweils einer Datei je Spalte.

-n[*Trennzeichen*[*Anzahl*]]
: Veranlaßt das Hinzufügen von Zeilennummern vor jeder Spaltenzeile oder bei paralleler Ausgabe vor jeder Zeile. Wenn ein *Trennzeichen* angegeben ist, wird dieses Zeichen anstelle des Tabulators an die Zeilennummer angehängt. Die optionale Angabe einer *Anzahl* begrenzt den Raum für die Zeilennummer auf diese Stellen.

-o *Weite*
: Setzt den linken Seitenrand auf die angegebene *Weite*.

-r Unterdrückt die Ausgabe von Warnungen, wenn eine Eingabedatei nicht geöffnet werden kann.

-s[*Trennzeichen*]
: Veranlaßt die Benutzung des angegebenen *Trennzeichens* (Default ist der Tabulator) anstelle der Leerzeichen zur Trennung der Ausgabespalten.

-t Unterdrückt die Ausgabe der Kopf- und Fußzeilen und der Füllzeilen am Seitenende.

-v Veranlaßt die Umwandlung nichtdruckbarer Zeichen in Oktalzahlen mit Backslash.

ps

ps [*Optionen*]

Gibt Informationen zu den aktiven Prozessen aus. Beachten Sie, daß Sie die Optionen nicht durch ein – einleiten müssen. Beim Vergleich der Menge ausgegebener Daten ergibt sich, daß -e mehr ausgibt als -a und -l mehr als -f.

Optionen

PIDs Es werden nur Daten zu den Prozessen mit den angegebenen Identifikationsnummern ausgegeben. Die Angabe auf der Kommandozeile erfolgt durch eine mit Kommata separierte Liste.

a	Es werden alle aktiven Prozesse aufgelistet.
c	Es wird das im **task_struct** verzeichnete Kommando angezeigt.
e	Die Variablen der Programmumgebung werden angezeigt.
f	Die Prozeßfamilien werden als Bäume dargestellt.
h	Die Ausgabe der Kopfzeile wird unterdrückt.
j	Jobs-Format.
l	Veranlaßt die Ausgabe eines langen Listings.
m	Memory-Format.
n	BenutzerIDs und WCHAN werden als Zahlenwerte ausgegeben.
r	Es werden nur die laufenden Prozesse angezeigt.
s	Signal-Format.

--sort:[+|-]*Key*[,[+|-]*Key*[,...]]

Wie O, jedoch zur Verwendung mit langen Schlüsselwörtern. Sehen Sie weiter im Abschnitt *Schlüsselwörter*.

t*TTY*	Es werden nur die auf dem angegebenen *TTY* laufenden Prozesse angezeigt.
u	Veranlaßt die Ausgabe des Benutzernamens und der Startzeit.
v	*vm*-Format.
w	Breites Format. Lange Zeilen werden nicht abgeschnitten.
x	Weitet die Anzeige auf solche Prozesse aus, die mit keinem Terminal verbunden sind.

O[+|-]*key*[,[+|-]*key*[,...]]

Sortiert die Ausgabeliste. Sehen Sie weiter im Abschnitt *Schlüsselwörter*.

+ Legt die normale Sortierrichtung für den Schlüssel fest.
– Kehrt die Sortierrichtung für den Schlüssel um.

S Die CPU-Zeit und die Seitenfehler der Kindprozesse werden den Eltern zugeschlagen.

Sortierschlüssel

c, cmd	Name der Programmdatei
C, cmdline	Gesamte Kommandozeile
f, flags	Flags
g, pgrp	Gruppenidentifikationsnummer des Prozesses
G, tpgid	Gruppenidentifikationsnummer des verbundenen Terminals
j, cutime	Kumulative Benutzerzeit
J, cstime	Kumulative Systemzeit
k, utime	Benutzerzeit
K, stime	Systemzeit
m, min_flt	Anzahl der „Minor Page Faults“
M, maj_flt	Anzahl der „Major Page Faults“
n, cmin_flt	Kumulierte Anzahl der „Minor Page Faults“
N, cmaj_flt	Kumulierte Anzahl der „Major Page Faults“
o, session	Session ID

→

ps
←

p, pid	Prozeßidentifikationsnummern
P, ppid	Prozeßidentifikationsnummern der Eltern
r, rss	Resident Set Size
R, resident	Resident Pages
s, size	Größe des benutzten Speichers
S, share	Anzahl der mit anderen Prozessen geteilten Speicherseiten
t, tty	Nummer des Terminals
T, start_time	Startzeit des Prozesses
U, uid	Benutzeridentifikationsnummer
u, user	Benutzername
v, vsize	Größe des benutzten virtuellen Speichers
y, priority	Schedulerpriorität des Prozesses

Felder

PRI	Schedulerpriorität des Prozesses. Eine höhere Zahl zeigt eine niedrigere Priorität an.
NI	Der Nice-Wert des Prozesses. Eine höhere Zahl bedeutet weniger CPU-Zeit.
SIZE	Die Größe des virtuellen Speicherbereiches.
RSS	Die „Resident Set Size" (die Größe des physikalischen Arbeitsspeichers) in Kilobyte.
WCHAN	Die Kernelfunktion, in der der Prozeß schläft.
STAT	Der Status des Prozesses.

R	Lauffähig
T	Gestoppt
D	Nicht unterbrechbarer Schlaf
S	Schlafend
Z	Zombie
W	Keine Speicherseiten belegt (zweites Feld)
N	Positiver Nice-Wert (drittes Feld)

TT	Verbundenes TTY
PAGEIN	Anzahl der „Major Page Faults"
TRS	Größe des Textsegments
SWAP	Größe des benutzten Swapbereichs
SHARE	Größe des mit anderen Prozessen geteilten Speicherbereichs

pwd

pwd

Gibt den vollständigen Pfadnamen des aktuellen Arbeitsverzeichnisses aus. Hinweis: Die eingebauten Versionen von **pwd** in den Bourne- und Korn-Shells und **dirs** in der C-Shell sind schneller. In der C-Shell können Sie folgenden Alias definieren:

```
alias pwd dirs -l
```

rcp [*Optionen*] *Datei1 Datei2*
rcp [*Optionen*] *Datei ... Verzeichnis*

rcp

Kopiert Dateien zwischen zwei Rechnern. Jede *Datei* oder das *Verzeichnis* ist entweder ein Name auf dem entfernten Rechner in der Form *Benutzer@Rechnername:Pfadname* oder ein lokaler Pfadname.

Optionen

- -p Erhält das Datum der letzten Änderung und den Zugriffsmodus der Quelldateien bei der Kopie.
- -r Wenn eine der Quellen ein Verzeichnis ist, werden alle darin enthaltenen Dateien und Verzeichnisse rekursiv in das Zielverzeichnis kopiert.

renice [*Priorität*] [*Optionen*] [*Ziel*]

renice

Kontrolliert die Schedulerpriorität verschiedener Prozesse, nachdem sie gestartet sind. Das Programm kann einzelne Prozesse, Prozeßgruppen und sämtliche Prozesse eines Benutzers beeinflussen. Der Superuser kann die Priorität der Prozesse anderer Benutzer verändern. Die *Priorität* für normale Benutzer muß zwischen 0 und dem Wert der Umgebungsvariablen PRIO_MAX (normalerweise 20) liegen. Höhere Zahlen machen den Prozeß „freundlicher" gegenüber anderen Prozessen. Der Superuser kann auch negative Prioritäten bis zum Wert von PRIO_MIN vergeben, um die Prozesse zu beschleunigen.

Optionen

- +*Zahl* Veranlaßt die Erhöhung der Prioritätszahl um *Zahl*.
- -*Zahl* Veranlaßt die Verringerung der Prioritätszahl um *Zahl*.
- -g Veranlaßt die Interpretation des angegebenen **Ziels** als Identifikationsnummer der Prozeßgruppe.
- -p Veranlaßt die Interpretation des angegebenen **Ziels** als Prozeßidenifikationsnummern (Voreinstellung).
- -u Veranlaßt die Interpretation des angegebenen **Ziels** als Benutzername.

reset

reset

Löscht den Bildschirminhalt und setzt das Terminal in den Einschaltzustand zurück.

rlogin *Rechnername* [*Optionen*]

rlogin

rlogin dient zum Einloggen in einen entfernten Rechner. Das Programm verbindet das Terminal des lokalen Rechners mit dem entfernten Zielrechner mit dem angegebenen *Rechnernamen*. Der Terminaltyp auf dem entfernten Rechner ist identisch mit dem Ihres lokalen Terminals. Die Fenstergröße wird ebenfalls vom entfernten Rechner übernommen, wenn der Server das erlaubt.

→

rlogin ←

Optionen

- -8 Erlaubt die Eingabe von 8-Bit-Daten zu jeder Zeit.
- -e*c* Veranlaßt die Verwendung des angegebenen Zeichens *c* als Fluchtsymbol (Voreinstellung ist ~).
- -d Schaltet in den Debug-Modus.
- -l *Benutzername*
  Veranlaßt die Verwendung des angegebenen *Benutzernamens* auf dem entfernten Rechner. Ohne diese Angabe wird der lokale Name des Benutzers verwendet.
- -E Unterdrückt die Interpretation irgendeines Fluchtsymbols.
- -L Erlaubt den Ablauf einer **rlogin**-Session ohne Nachbearbeitung der Ausgabe (**litout**-Modus).

---

rm

rm [*Optionen*] *Dateien*

Löscht eine oder mehr *Dateien*. Um eine Datei löschen zu können, brauchen Sie Schreibberechtigung für das Verzeichnis, in dem die Datei enthalten ist, Sie brauchen aber kein Schreibrecht für die Datei selbst. Wenn Sie kein Schreibrecht für die Datei haben, müssen Sie den Löschbefehl vor der Ausführung bestätigen (**y** oder **n**).

Optionen

- -d, --**directory**
  Löscht auch Verzeichnisse, selbst wenn diese nicht leer sind. Das Programm benutzt dazu **unlink** anstelle von **rmdir**. Diese Option kann nur mit Superuser-Rechten benutzt werden.
- -f, --**force**
  Löscht schreibgeschützte Dateien ohne zusätzliche Bestätigung.
- -i, --**interactive**
  Erwartet vor der Löschung jeder einzelnen Datei eine interaktive Bestätigung (**y** zum Löschen) oder eine Ablehnung (**n** für Erhalten).
- -r, -R, --**recursive**
  Wenn anstelle einer *Datei* ein Verzeichnis auf der Kommandozeile angegeben ist, werden das Verzeichnis und sein gesamter Inhalt mit allen Unterverzeichnissen rekursiv gelöscht. Seien Sie gewarnt: Diese Option kann gefährlich sein.
- -- Kennzeichnet das Ende der Optionen. Benutzen Sie diese Notation, wenn Sie einen Dateinamen übergeben müssen, der mit einem - beginnt.

---

rmdir

rmdir [*Optionen*] *Verzeichnisse*

Löscht die angegebenen *Verzeichnisse* (nicht deren Inhalt). *Verzeichnisse* werden aus ihren Überverzeichnissen gelöscht und müssen leer sein. (Wenn das nicht der Fall ist, können Sie statt dessen **rm -r** benutzen.) Sehen Sie auch **mkdir**.

Option

-p, --parents
: Löscht *Verzeichnisse* und alle übergeordneten Verzeichnisse des Pfades, die als Resultat der Löschung leer werden. Diese Option dient zur Löschung leerer Verzeichnisbäume.

rpcinfo [*Optionen*] [*Version*]

rpcinfo (NFS-/NIS-Programm) testet einen Server und gibt Informationen darüber aus. Programme können als Name oder Nummer angegeben werden. Wenn die *Version* unbekannt ist, geben Sie als Versionsnummer 0 an, um **rpcinfo** zu veranlassen, die Version zu erraten.

Optionen

-b *Programm Version*
: Veranlaßt einen RPC Broadcast (unter Verwendung des User Datagram Protocols) nach dem angegebenen *Programm* mit der angegebenen *Version*. Es wird der Host ausgegeben, der als erster antwortet.

-d *Programm Version*
: Mit Superuser-Rechten ausgeführt, löscht diese Option die Registrierung des angegebenen *Programms*.

-n *Portnummer*
: Veranlaßt **rpcinfo**, die RPC-Anfragen auf der angegebenen *Portnummer* durchzuführen.

-p [*Host*]
: Gibt eine Liste aller beim angegebenen *Host* registrierten RPC-Programme aus. (Wenn kein *Host* angegeben ist, wird die Anfrage an den lokalen Rechner gerichtet, dessen Name durch das Programm **$hostname** ermittelt wird.)

-u *Host Programm*
: Veranlaßt den Versuch, einen RPC-Call mit UDP zum angegebenen *Programm* auf dem *Host* durchzuführen. Es werden Meldungen ausgegeben, ob der Versuch erfolgreich war.

-t *Host Programm*
: Veranlaßt den Versuch, einen RPC-Call mit TCP zum angegebenen *Programm* auf dem *Host* durchzuführen. Es werden Meldungen ausgegeben, ob der Versuch erfolgreich war.

rsh [*Optionen*] *Host* [*Kommando*]

Führt ein *Kommando* auf einem entfernten Rechner aus oder startet eine interaktive Shell mit **rlogin**, wenn kein Kommando angegeben ist.

Optionen

-d
: Schaltet Socket-Debugging ein.

-l *Benutzername*
: Veranlaßt die Verwendung des angegebenen *Benutzernamens*

→

rsh ←

auf dem entfernten Rechner. Ohne diese Angabe wird der lokale Name des Benutzers verwendet.

-n
: Veranlaßt die Umlenkung der Standardeingabe für **rsh** auf das Spezialdevice */dev/null*. (Diese Option sollte verwendet werden, wenn ein **rsh**-Prozeß vom Prompt der **csh** in den Hintergrund geschickt wird, um die Eingabe vom Terminal weg zu lenken.)

---

rstat

rstat *Host*

Gibt eine zusammenfassende Statusinformation des angegebenen *hosts* an: Uptime, durchschnittliche Last und aktuelle Zeit.

---

run-parts

run-parts [*Optionen*] *Verzeichnis*

Führt Skriptdateien in den angegebenen **Verzeichnissen** aus. Alle Dateien, deren Namen ausschließlich aus alphanumerischen Zeichen, Unterstrichen und Bindestrichen bestehen, werden als Skriptdateien angesehen. Alle übrigen Dateien werden ignoriert. Die Skriptdateien werden in der Reihenfolge ihrer lexikalischen Ordnung ausgeführt.

Optionen

--
: Ende der Optionen; alle Argumente nach dieser Option werden als Verzeichnisname interpretiert, auch wenn deren Name mit einem – beginnt.

--test
: Testmodus: Es wird Information ausgegeben, welche Skriptdateien ausgeführt würden, ohne daß das tatsächlich geschieht.

--umask=*umask*
: Setzt die Erzeugungsmaske auf den angegebenen Wert (Voreinstellung ist 022).

---

ruptime

ruptime [*Optionen*]

Zeigt den Host-Status der Rechner im lokalen Netz an. Die Statuszeilen werden aus den minütlichen Broadcast-Paketen aller Rechner im Netz ermittelt. Rechner, von denen elf Minuten kein aktiver Status ermittelt werden konnte, werden als down angezeigt.

Optionen

-l
: Veranlaßt die Sortierung nach durchschnittlicher Last.

-r
: Veranlaßt die Umkehrung der Reihenfolge.

-t
: Veranlaßt die Sortierung nach Uptime.

-u
: Veranlaßt die Sortierung nach der Anzahl aktiver Benutzer.

---

rusers

rusers [*Optionen*] [*Host . . .*]

Gibt eine Liste der auf einem *Host* eingeloggten Benutzer aus (RPC-Version). **rusers** erzeugt eine Liste, ähnlich wie das Programm **who**. Wenn eine Liste von *Hosts* angegeben ist, werden diese aktiven Benutzer dieser Rechner

aufgelistet. Ein entfernter Rechner gibt die Information nur weiter, wenn dort der **rusersd**-Dämon läuft. Broadcasting über Gateways ist nicht möglich.

Optionen

- `-a` Erzeugt eine Ausgabe auch für Rechner, auf denen kein Benutzer eingeloggt ist.
- `-l` Erzeugt ein ausführlicheres Listing im Stil von **who**.

rwall *Hostname* [*Datei*]

Schreibt Mitteilungen an alle Benutzer eines Netzwerkes. **rwall** liest die Mitteilung von der Standardeingabe bis zum Zeichen *EOF*. Dann schickt das Programm die Mitteilung zusammen mit der einleitenden Zeile `Broadcast Message ...` an alle auf dem in der Kommandozeile angegebenen Host eingeloggten Benutzer. Wenn eine *Datei* angegeben ist, wird deren Inhalt gesendet und nicht von der Standardeingabe gelesen.

script [*Optionen*] [*Datei*]

Kopiert die Terminalausgabe in die Datei *typescript* oder in die auf der Kommandozeile angegebene *Datei*. Die Mitschrift wird beendet, wenn die Shell verlassen wird.

Option

- `-a` Die Daten werden an eine existierende Datei angehängt, anstatt sie zu überschreiben.

sed [*Optionen*] [*Kommando*] [*Dateien*]

sed ist ein „Datenstromeditor" zur Bearbeitung einer Liste von *Dateien* ohne interaktive Beteiligung des Benutzers. Lesen Sie Kapitel 9, um weitere Informationen zu erhalten.

Optionen

-e '*Anweisungen*', **--expression=**'*Anweisungen*'
: Führt die angegebenen *Anweisungen* an den Dateien aus.

-f *Skript*, **--file=***Skript*
: Führt die in *Skript* aufgeführten Anweisungen aus.

-n, **--quiet**, **--silent**
: Unterdrückt die Ausgabe unbearbeiteter Zeilen.

setfdprm [*Optionen*] *Gerätedatei* [*Name*]

Stellt die Diskettenparameter zur Konfiguration des Floppylaufwerkes ein.

Optionen

-c *Gerätedatei*
: Löscht die Parameter der angegebenen *Gerätedatei*.

→

setfdprm ←

-n *Gerätedatei*
: Unterdrückt die Formatanzeige bei der automatischen Erkennung für die angegebene *Gerätedatei.*

-p *Gerätedatei* [*Name* | *Parameter*]
: Lädt einen neuen Parametersatz mit dem angegebenen *Namen* zur automatischen Formaterkennung aus der Datei */etc/fdprm* oder lädt die direkt auf der Kommandozeile angegebenen *Parameter.* Sie können die Werte für **dev**, **size**, **sect**, **heads**, **tracks**, **stretch**, **gap**, **rate**, **spec1** und **fmt_gap** angeben.

-y *Gerätedatei*
: Veranlaßt den Floppytreiber, bei der automatischen Erkennung das gefundene Format als Kernelnachricht anzuzeigen.

---

sh

sh [*Optionen*] [*Datei* [*Argumente*]]

Die Standardshell für UNIX, ein Kommandozeileninterpreter, in dem alle weiteren Kommandos eingegeben werden. Unter Linux ist das in der Regel ein anderer Name für die **bash**-Shell. Siehe Kapitel 3.

---

showmount

showmount [*Optionen*] [*Host*]

showmount (NFS-/NIS-Programm) zeigt Informationen über den NFS-Server. Diese Information wird vom Dämonprogramm **mountd** auf dem befragten *Host* geliefert. Die Voreinstellung für *Host* ist der vom Programm **hostname** gelieferte lokale Rechnername. Ohne Optionen werden alle Clients angezeigt, die Verzeichnisse des lokalen Rechners gemountet haben.

Optionen

-a, --all
: Gibt alle entfernten Mounts im Format

  Hostname:Verzeichnis

  aus, wobei *Hostname* der Name des Clients und *Verzeichnis* der Quellpfadname des gemounteten Verzeichnisses ist.

-d, --directories
: Gibt eine Liste der von entfernten Clients gemounteten Verzeichnisse aus.

-e, --exports
: Gibt eine Liste der exportierten Dateisysteme aus.

--no-headers
: Unterdrückt die Ausgabe der Kopfzeile.

---

sleep

sleep *Dauer* [*Einheit*]

Wartet die angegebene *Dauer*, bevor ein anderes Kommando ausgeführt werden kann. Die Voreinstellung für *Einheit* ist Sekunden.

Zeit	Einheit
s	Sekunden

Zeit	Einheit
m	Minuten
h	Stunden
d	Tage

sort [*Optionen*] [*Dateien*]

Sortiert die Zeilen der angegebenen *Dateien*. Vergleicht bestimmte Felder jeweils zweier Zeilen oder, falls keine Felder bestimmt sind, die gesamten Zeilen paarweise in der systemspezifischen Ordnung. Siehe auch **uniq**, **comm** und **join**.

Optionen

-b
: Führende Tabs und Leerzeichen werden ignoriert.

-c
: Überprüft, ob die *Dateien* bereits sortiert sind. In diesem Fall wird keine Ausgabe erzeugt.

-d
: Veranlaßt die Sortierung in lexikalischer Ordnung.

-f
: „Fold"— ignoriert Groß- und Kleinschreibung.

-i
: Ignoriert nichtdruckbare Zeichen (solche außerhalb des Bereiches ASCII 040–176).

-m
: Fügt mehrere sortierte Eingabedateien zusammen.

-n
: Veranlaßt die Sortierung in arithmetischer Ordnung.

-o*Datei*
: Veranlaßt die Ausgabe in die angegebene *Datei*.

-r
: Veranlaßt die Umkehrung der Ordnung.

-t*c*
: Bestimmt das angegebene Zeichen *c* zum Feldtrenner des Sortierschlüssels (Voreinstellung ist Tab).

-u
: Identische Eingabezeilen erscheinen nur einmal (unique) in der Ausgabe.

+*n* [-*m*]
: Bestimmt die Zeichen zwischen *n* (ausschließlich) und *m* (einschließlich) zum Sortierschlüssel; wenn *m* fehlt, werden alle Zeichen bis zum Zeilenende zum Schlüssel. Positionen werden in der Form *a.b* angegeben, das bedeutet Zeichen Nummer *b* von Feld Nummer *a*. Wenn kein *b*-Teil angegeben ist, wird nach dem ersten Zeichen im Feld sortiert.

-k *n*[,*m*]
: Ähnlich wie +. Die ersten *n*–1 werden ignoriert und die Felder *n* bis *m*–1 zur Sortierung verwendet. Die Felder werden hier mit 1 beginnend numeriert.

-M
: Veranlaßt die Interpretation der ersten drei Buchstaben jeder Zeile als Monatskürzel (JAN, FEB etc). Die Sortierung erfolgt in der Reihenfolge JAN < FEB < *ungültiger Monatsname*.

-T *Verzeichnis*
: Bestimmt das angegebene *Verzeichnis* zur Benutzung für temporäre Dateien.

Beispiele

Erzeugt eine Dateiliste mit abfallender Anzahl von Zeilen:

```
wc -l * | sort -r
```

→

sort ←

Bringt eine Wortliste in die lexikalische Ordnung, entfernt Duplikate und gibt die Häufigkeit der einzelnen Wörter aus:

```
sort -fd wordlist | uniq -c
```

Sortiert die Paßwortdatei in numerischer Ordnung nach dem dritten Feld (Benutzer-ID):

```
sort +2n -t: /etc/passwd
```

---

split

split [*Option*] [*Quelldatei*] [*Zielpräfix*]

split teilt eine *Quelldatei* in Segmente mit einer bestimmten Anzahl von Zeilen. Die *Quelldatei* bleibt unverändert, und die Produkte landen in Ausgabedateien mit den Namen *Zielpräfix***aa**, *Zielpräfix***ab** und so weiter (Voreinstellung ist **xaa**, **xab** etc.). Das Programm liest von der Standardeingabe, wenn keine *Quelldatei* angegeben ist oder statt dessen ein - übergeben wird. Sehen Sie auch **csplit**.

Optionen

-*n*, **-l** *n*, **--lines=***n*
: Teilt die *Quelldatei* in Segmente mit jeweils *n* Zeilen (Voreinstellung ist 1000).

-b *n*[**bkm**], **--bytes=***n*[**bkm**]
: Teilt die *Quelldatei* in Segmente zu jeweils *n* Bytes bzw. der angegebenen Zahl folgender Einheiten:

b	512 Bytes
k	1 Kilobyte
m	1 Megabyte

-C *Bytes*[**bkm**], **--line-bytes=***Bytes*[**bkm**]
: Teilt die *Quelldatei* in Segmente mit maximal *n* Bytes. Es wird zusätzlich sichergestellt, daß nur vollständige Zeilen eingefügt werden.

-
: Veranlaßt das Lesen der Quelldaten von der Standardeingabe.

Beispiele

Teile *bigfile* in Segmente mit jeweils 1000 Zeilen:

```
split bigfile
```

Füge vier Dateien zusammen und teile sie wieder in Dateien zu jeweils zehn Zeilen mit den Namen *new.aa*, *new.ab* etc. Beachten Sie, daß ohne das – in der Kommandozeile **new.** als nichtexistierender Dateiname behandelt würde.

```
cat list[1-4] | split -10 - new.
```

---

stty

stty [*Optionen*] [*Modi*]

Setzt die Terminal-I/O-Optionen für das aktuelle Eingabegerät. Ohne Optionen zeigt das Programm die aktiven Einstellungen an, sofern sie von den mit **stty sane** erzeugten Standardeinstellungen abweichen. Dabei steht ^

für ein CONTROL-Zeichen und ^` für einen leeren Eintrag. Die meisten Modi können umgekehrt oder abgeschaltet werden, indem ein optionales – vorangestellt wird (durch eckige Klammern dargestellt). Die entsprechende Beschreibung wird ebenfalls in eckigen Klammern dargestellt. Einige Argumente benutzen von POSIX abweichende Erweiterungen; sie sind durch ein * gekennzeichnet.

```
stty [Optionen] [Modi] [< Device]
```

Optionen

-a, --all	Zeigt alle veränderbaren Einstellungen an.
-g	Zeigt die Einstellungen in hexadezimaler Form.

Steuermodi

n	Setzt die Übertragungsgeschwindigkeit zum Terminal auf *n* (z.B. 2400).
[-]**clocal**	Schaltet die Modemsteuerung [ein] aus.
[-]**cread**	Schaltet den Empfänger ein [aus].
cs*bits*	Wählt die Zahl der Bits pro Zeichen aus (5 - 8).
[-]**cstopb**	[Ein] Zwei Stop-Bits pro gesendetes Zeichen.
[-]**hup**	Trennt die (Modem-)Verbindung nach dem letzten Schließen [nicht].
[-]**hupcl**	Wie oben.
ispeed *n*	Setzt Terminalempfangsgeschwindigkeit auf *n*.
ospeed *n*	Setzt Terminalsendegeschwindigkeit auf *n*.
[-]**parenb**	Setzt die Paritätserzeugung und -prüfung [nicht].
[-]**parodd**	Nutzt [gerade] ungerade Parität.
[-]**crtscts***	Setzt RTS-/CTS-Handshaking [nicht].

Datenflußkontrolle

Die folgenden Modi zur Datenflußkontrolle sind durch Kombination der Flags für *ortsfl*, *ctsflow* und *rtsflow* möglich:

Flag-Einstellung	*Datenflußmodus*
ortsfl rtsflow ctsflow	Unidirektionale Flußkontrolle
ortsfl rtsflow -ctsflow	RTS wird bei Sendebereitschaft gesetzt
ortsfl -rtsflow ctsflow	Ohne Effekt
ortsfl -rtsflow -ctsflow	Bidirektionale Flußkontrolle
-ortsfl rtsflow ctsflow	Bidirektionale Flußkontrolle
-ortsfl rtsflow -ctsflow	Ohne Effekt
-ortsfl -rtsflow ctsflow	Datenübertragung wird bei fallendem CTS angehalten
-ortsfl -rtsflow -ctsflow	Schaltet Hardware-Flußkontrolle ab

Eingabemodi

[-]**brkint**	Signalisiert INTR bei einem Break [nicht].
[-]**icrnl**	Bildet bei der Eingabe CR [nicht] auf NL ab.

→

stty
←

[-]**ignbrk**	Ignoriert Break bei Eingabe [nicht].
[-]**igncr**	Ignoriert CR bei Eingabe [nicht].
[-]**ignpar**	Ignoriert Paritätsfehler [nicht].
[-]**inlcr**	Bildet bei der Eingabe NL [nicht] auf CR ab.
[-]**inpck**	Schaltet die Eingabeparitätsprüfung [nicht] ein.
[-]**istrip**	Verkürzt die Eingabezeichen [nicht] auf 7 Bits.
[-]**iuclc***	Wandelt eingehende Großbuchstaben [nicht] in Kleinbuchstaben um.
[-]**ixany***	Startet mit jedem Zeichen [XON] die Ausgabe.
[-]**ixoff** [-]**tandem**	Sendet [keine] START-/STOP-Zeichen, wenn die Eingabewarteschlange fast leer/voll ist.
[-]**ixon**	Schaltet die START-/STOP-Ausgangskontrolle [nicht].
[-]**parmrk**	Markiert Paritätsfehler [nicht].
[-]**imaxbel***	Wenn der Eingabepuffer zu voll ist, um ein weiteres Zeichen zu empfangen, [wird der Eingabepuffer geleert] wird ein akustisches Signal ausgegeben und der Eingabepuffer nicht geleert.

Ausgabemodi

bs*n*	Wählt Verzögerungsart für Rückschrittzeichen (0 oder 1).
cr*n*	Wählt Verzögerungsart für Wagenrücklauf (0-3).
ff*n*	Wählt Verzögerungsart für Seitenvorschübe (0 oder 1).
nl*n*	Wählt Verzögerungsart für Zeilenvorschübe (0 oder 1).
tab*n*	Wählt Verzögerungsart für horizontale Tabulatoren (0-3).
vt*n*	Wählt die Verzögerungsart für vertikale Tabulatoren (0 oder 1).
[-]**ocrnl***	Bildet bei der Ausgabe NL [nicht] auf CR-NL ab.
[-]**ofdel***	Setzt die Füllzeichen auf [NULL] DEL.
[-]**ofill***	Verzögert die Ausgabe mit [Warten] Füllzeichen.
[-]**olcuc***	Wandelt ausgehende Kleinbuchstaben [nicht] in Großbuchstaben um.
[-]**onlcr***	Bildet bei der Ausgabe NL [nicht] auf CR-NL ab.
[-]**onlret***	Auf dem Terminal führt NL [nicht] die CR-Funktion aus.
[-]**onocr***	Gibt in Spalte 0 [keine] CRs aus.
[-]**opost**	Bearbeitet die Ausgabe [nicht] nach; ignoriert alle anderen Ausgabemodi.

Lokale Modi

[-]**echo**	Wiederholt die gelesenen Zeichen [nicht].
[-]**echoe**, [-]**crterase**	Stellt [nicht] ERASE-Zeichen als Rückschritt-Leerzeichen-Rückschritt-Zeichenfolge dar.
[-]**echok**	Gibt [kein] NL nach KILL-Zeichen aus.
[-]**echonl**	Ein NL wird im kanonischen Modus [nicht] reflektiert, auch wenn das Echo für andere Zeichen abgeschaltet ist.

[-]icanon Schaltet die kanonische Eingabe zur Sonderbehandlung von ERASE, KILL, WERASE und RPRINT [nicht] ein.

[-]iexten Schaltet die erweiterten Funktionen für Eingabedaten [nicht] ein.

[-]isig Schaltet die Zeichenprüfung hinsichtlich INTR, SUSPEND und QUIT [nicht] ein.

[-]noflsh Schaltet die Pufferausgabe nach INTR oder QUIT [nicht] ab.

[-]tostop* Sendet [kein] SIGTTOU, wenn Hintergrundprozesse auf ein Terminal schreiben.

[-]xcase* Ändert bei der lokalen Ausgabe die Groß- und Kleinschreibung

[-]echoprt, [-]prterase*
Beim Löschen werden Zeichen [nicht] in \ und / eingeschlossen reflektiert.

[-]echoctl. [-]ctlecho*
Reflektiert Kontrollzeichen [nicht] als *^Zeichen* (z.B. ^Z).

[-]echoke [-]crtkill*
Löschzeichen wie durch die Einstellungen **echoprt** und **echoe** definiert.

Steuerungszuweisungen

ctrl-char c Setzt das Steuerzeichen auf *c*. *Steuerzeichen* sind: **dsusp** (Eingabepuffer-Flush und sendet Stop), **eof**, **eol**, **eol2** (alternatives End-of-line), **erase**, **intr**, **lnext** (behandelt nächstes Zeichen uninterpretiert), **kill**, **rprnt** (schreibt Zeile neu), **quit**, **start**, **stop**, **susp**, **swtch** oder **werase** (löscht letztes Zeichen). *c* kann ein buchstäbliches Kontrollzeichen, ein Zeichen in Caret-Notation (z.B. ^Z), eine Hexadezimalzahl (muß mit 0x beginnen), eine Oktalzahl (muß mit 0 beginnen) oder eine Dezimalzahl sein. Sie schalten ein Kontrollzeichen ab, indem Sie den Wert ^- oder **undef** zuweisen.

min *n* Setzt die Minimalzahl von Zeichen, die einen Lesevorgang im **-icanon**-Modus abschließen, bevor der Timer abgelaufen ist.

time *n* Setzt die Anzahl der Zehntelsekunden, nach denen der Timer im **-icanon**-Modus abläuft, bevor die in **min** festgelegte Anzahl Zeichen gelesen wurde.

line *i* Setzt das Schnittstellenprotokoll (Line-Discipline) auf *i* (1–126).

Kombinationsmodi

cooked Entspricht **-raw**.

[-]evenp [-]parity
Entspricht [–]**parenb** und **cs**[8]7.

[-]parity Entspricht [-]**parenb** und **cs**[8]7.

→

stty ←

ek	Setzt die Zeichen ERASE und KILL auf **CTRL-h** und **CTRL-u**.
[-]**lcase**	Setzt [Hebt] **xcase**, **iuclc** und **olcuc** [auf].
[-]**LCASE**	Entspricht [-]**lcase**.
[-]**nl**	Setzt [Hebt] **icrnl** und **onlcr** [auf]. **-nl** hebt darüber hinaus **inlcr**, **igncr**, **ocrnl** und **onlret** auf.
[-]**oddp**	Entspricht [-]**parenb**, [-]**parodd** und **cs7**[**8**].
[-]**raw**	Schaltet die raw-Ein- und -Ausgabe (nicht ERASE, KILL, INTR, QUIT, EOT, SWITCH oder Ausgabenachbearbeitung) [nicht] ein.
sane	Setzt alle Modi auf sinnvolle Werte zurück.
[-]**tabs***	[Wandelt die Tabulatoren in Leerzeichen um.] Behält die Tabulatoren bei der Ausgabe bei.
[-]**cbreak**	Entspricht **-icanon**.
[-]**pass8**	Entspricht **-parenb -istrip cs8**.
[-]**litout**	Entspricht **-parenb -istrip cs8**.
[-]**decctlq***	Entspricht **-ixany**.
crt	Entspricht **echoe echoctl echoke**.
dec	Entspricht **echoe echoctl echoke -ixany**. Zusätzlich wird INTERRUPT zu ^C, ERASE zu DEL und KILL zu ^U.

Spezialeinstellungen

ispeed *speed*	Bestimmt die Eingabegeschwindigkeit.
ospeed *speed*	Bestimmt die Ausgabegeschwindigkeit.
rows *rows**	Bestimmt die Anzahl der Zeilen.
cols *columns*, **columns** *columns**	Bestimmt die Anzahl der Spalten.
size*	Gibt die aktuelle Einstellung für Zeilen- und Spaltenzahl aus.
speed	Gibt die aktuelle Terminalgeschwindigkeit aus.

su

su [*Option*] [*Benutzer*] [*Shellargumente*]

Erzeugt eine Shell mit der effektiven Benutzeridentifikation eines anderen *Benutzers*. Wenn kein *Benutzer* angegeben ist, wird eine Shell für den Superuser (root) erzeugt (Sie werden also Superuser). Geben Sie das Zeichen für *EOF*, um die Shell wieder zu verlassen. Sie können die Shell mit bestimmten Optionen starten, indem Sie die entsprechenden *Shellargumente* übergeben (wenn Sie beispielsweise die Shell **sh** verwenden können Sie **-c** *Kommando* angeben, um dieses *Kommando* direkt auszuführen.)

Optionen

-, **-l**, **--login**
: Veranlaßt die Durchführung einer normalen Login-Sequenz (z.B. wird die Laufzeitumgebung des entsprechenen Benutzers initialisiert).

-c *Kommando*, --**command**=*Kommando*
Führt das angegebene *Kommando* in der neuen Shell aus und terminiert sofort nach dessen Beendigung. Wenn das *Kommando* aus mehr als einem Wort besteht, sollte es in Anführungsstrichen eingeschlossen werden — zum Beispiel:

```
su -c 'find / -name \*.c -print' nobody
```

-f, --**fast**
Veranlaßt den Aufruf der Shell mit der Option -f. Bei der **csh** und der **tcsh** unterdrückt dies das Lesen der Datei *.cshrc*. Bei der **bash** unterdrückt das die Dateinamenserweiterung.

-m, -p, --**preserve-environment**
Bewahrt die originalen Werte der Umgebungsvariablen HOME, USER, LOGNAME und SHELL. Führt $SHELL anstelle des in */etc/passwd* festgelegten Programms aus, wenn $SHELL nicht eingeschränkt ist.

-s *Shell*, --**shell** *Shell*
Führt das Programm *Shell* anstelle des in */etc/passwd* festgelegten Programms aus, wenn *Shell* nicht eingeschränkt ist.

---

symlinks [*Optionen*] *Verzeichnisse*

symlinks

Gibt eine Liste der symbolischen Links mit zusätzlichen Informationen aus. Die angegebenen *Verzeichnisse* werden nach symbolischen Links durchsucht und mit den Attributen **relative**, **absolute**, **dangling** oder **other_fs** belegt.

Optionen

-c
Wandelt absolute Links in relative Links um, es sei denn, sie zeigen auf ein anderes Dateisystem.

-r
Durchläuft rekursiv alle Unterverzeichnisse und sucht überall nach symbolischen Links.

-v
Zeigt auch relative Links an.

---

systat [*Optionen*] *Host*

systat

Fragt Statusinformationen vom Netzwerkrechner *Host* ab. Zuerst wird versucht, den Service **systat** zu benutzen; wenn das kein Ergebnis liefert, werden die Services **netstat** und **daytime** verwendet.

Optionen

-n, --**netstat**
Veranlaßt die Benutzung des **netstat**-Services.

-p *Port*, --**port** *Port*
Richtet die Anfrage an den angegebenen *Port* des *Hosts*.

-s, --**systat**
Veranlaßt die Benutzung des **systat**-Services.

-t, --**time**
Veranlaßt die Benutzung des **daytime**-Services.

tac

tac [*Optionen*] [*Datei*]

Das Programm **tac** ist nach dem bekannteren **cat** benannt und gibt Dateien rückwärts aus. Ohne Angabe einer *Datei* oder mit einem - an deren Stelle liest das Programm von der Standardeingabe. **tac** arbeitet feldorientiert und gibt alle Felder in umgekehrter Reihenfolge aus. Als Feldtrenner ist das Zeilenende voreingestellt, die letzte Zeile wird also zuerst ausgegeben.

Optionen

-b, --**before**
: Legt fest, daß der Feldtrenner vor dem Feld steht, das er begrenzt.

-r, --**regex**
: Ermöglicht die Verwendung von regulären Ausdrücken bei der Angabe des Feldtrenners.

-s *Feldtrenner*, --**separator**=*Feldtrenner*
: Legt einen neuen *Feldtrenner* fest (Voreinstellung ist das Zeilenende).

tail

tail [*Optionen*] [*Datei*]

Gibt die letzten zehn Zeilen der angegebenen *Datei* (oder der Standardeingabe, wenn an deren Stelle ein - erscheint) aus.

Optionen

-*n*[*k*]
: Beginnt die Ausgabe an der *n*ten Stelle vom Ende der Datei an. Durch *k* wird angegeben, wie die Stellen gezählt werden: **l** steht für Zeilen (Lines, die Voreinstellung), **b** für Blöcke (512 Bytes) oder **c** für Zeichen (character).

+*n*[*k*]
: Wie -*n*, die Stellen werden jedoch vom Anfang der Datei an gezählt.

-c *Anzahl*{**bkm**}, --**bytes** *Anzahl*{**bkm**}
: Gibt die angegebene *Anzahl* Bytes vom Ende der Datei her aus. Eine abweichende Blockgröße kann durch Anhängen der folgenden Buchstaben erreicht werden:

b	512 Bytes
k	1 Kilobyte
m	1 Megabyte

-f
: Veranlaßt das Programm, nicht am Dateiende abzubrechen, sondern mit der Ausgabe zu folgen, wenn die Datei wächst. Sie beenden das Programm durch die Tastenkombination **CTRL-C**.

-n *num*, --**lines** *num*
: Gibt die letzten *num* Zeilen aus.

-q, --**quiet**, --**silent**
: Unterdrückt die Ausgabe der Dateinamen als Kopfzeile.

Beispiele

Zeigt die letzten 20 Zeilen, in denen die Zeichenkette **.Ah** vorkommt:

```
grep '\.Ah' file | tail -20
```

Zeigt die letzten zehn Zeichen der Shellvariablen **name**:

```
echo "$name" | tail -c
```

Gibt die letzten zwei Blöcke der Datei **bigfile** aus:

```
tail -2b bigfile
```

talk

talk *Person* [*TTY*]

Das Programm **talk** ermöglicht den Dialog mit einem Benutzer an einem anderen Terminal. Die angesprochene *Person* ist entweder der Loginname von einem Benutzer des lokalen Rechners oder eine Adresse der Form *user@host* auf einem entfernten Rechner. Um mit einem Benutzer zu kommunizieren, der mehrfach eingeloggt ist, geben Sie zusätzlich den Namen des *TTY* an, um das gewünschte Terminal zu bezeichnen. Sobald die Verbindung hergestellt ist, können die Beteiligten simultan schreiben und die Ausgabe gleichzeitig in den geteilten Terminalfenstern lesen. Um den Bildschirminhalt neu aufzubauen, geben Sie die Tastenkombination **CTRL-L** ein. Um das Programm zu beenden, geben Sie die Tastenkombination zur Unterbrechung ein (normalerweise **CTRL-C**); **talk** stellt den Bilschirminhalt wieder so her, wie es ihn beim Aufruf vorgefunden hat.

tcsh

tcsh [*Optionen*] [*Datei* [*Argumente*]]

Eine erweiterte Version der C-Shell, ein Kommandozeileninterpreter, in dem alle weiteren Kommandos eingegeben werden. Um weitere Informationen zu erhalten, lesen Sie Kapitel 4.

tee

tee [*Option*] *Datei*

Liest die Ausgabe eines anderen Programms und kopiert sie einerseits auf die Standardausgabe und andererseits in die angegebene *Datei*.

Option

-a Die Daten werden an die *Datei* angehängt, anstatt sie zu überschreiben.

Beispiel

`ls -l | tee savefile` *Zeigt ein Listing und speichert es für später*

telnet

telnet [*Optionen*] [*Host* [*Port*]]

Das Programm **telnet** ist die Benutzerschnittstelle, durch welche die Kommunikation mit einem Netzwerkrechner mittels des TELNET-Protokolls abgewickelt wird. Wenn **telnet** ohne die Angabe eines *Hosts* aufgerufen

→

telnet
←

wird, startet es im Befehlsmodus, was durch die Eingabeaufforderung `telnet>` angezeigt wird. Die ausführbaren Befehle werden nach der Beschreibung der Optionsliste erklärt. Wenn das Programm mit Argumenten aufgerufen wird, wird der Befehl **open** (die Beschreibung folgt) mit diesen Argumenten ausgeführt. *Host* bezeichnet einen offiziellen Rechnernamen, und die Angabe eines *Ports* ermöglicht den Aufbau der Verbindung mit einem anderen als dem üblichen TELNET-Port.

Optionen

-a
: Veranlaßt die Durchführung einer automatischen Login-Prozedur auf dem entfernten System.

-d
: Schaltet das Debugging auf Socket-Ebene ein.

-e [*Fluchtzeichen*]
: Bestimmt das angegebene *Fluchtzeichen*, mit dem die Telnet-Session unterbrochen werden kann. Wenn kein Zeichen nach dem Optionsschalter angegeben ist, wird das vordefinierte Fluchtzeichen gelöscht.

-l *Benutzername*
: Wenn die Verbindung zu einem entfernten System aufgebaut wird und das System die Option ENVIRON unterstützt, wird der angegebene *Benutzername* als USER für die Login-Prozedur übergeben.

-n *Tracefile*
: Öffnet das angegebene *Tracefile* und schreibt Informationen zu einer TN3270-Verbindung mit, sofern diese Variante unterstützt wird.

Befehle

z
: Suspendiert **Telnet**.

! [*Systemkommando*]
: Führt das *Systemkommando* in einer Subshell auf dem lokalen Rechner aus. Wenn kein *Systemkommando* angegeben ist, wird eine interaktive Shell gestartet.

? [*Befehl*]
: Gibt einen Hilfstext zum angegebenen *Befehl* aus. Ohne Argument wird eine Hilfsübersicht ausgegeben.

close
: Schließt eine laufende TELNET-Session und verbleibt im Befehlsmodus.

display *Argument* ...
: Zeigt die Einstellungen für alle oder für die als *Argument* angegebenen **set**- und **toggle**-Werte an.

mode [*Typ*]
: Der Status, in dem sich die TELNET-Session befindet, kann durch Angabe der folgenden *Typen* beeinflußt werden:

  ?
  : Gibt eine kurze Hilfe zur Benutzung des **mode**-Befehls.

  character
  : Schaltet den TELNET LINEMODE ab oder, wenn der entfernte Rechner diese Option nicht unterstützt, wechselt in den „Character at a time"-Modus.

[-]edit Schaltet den EDIT-Modus von TELNET LINEMODE an bzw. ab.

[-]isig Schaltet den TRAPSIG-Modus von TELNET LINEMODE an bzw. ab.

line Schaltet den TELNET LINEMODE ein oder, wenn der entfernte Rechner diese Option nicht unterstützt, wechselt in den „Old line by line"-Modus.

[-]softtabs Schaltet den SOFT_TAB-Modus von TELNET LINE-MODE an bzw. ab.

[-]litecho Schaltet den LIT_ECHO-Modus an bzw. ab.

open [-l *Benutzer*] *Host* [*Port*]

Öffnet eine Verbindung zum angegebenen *Host.* Wenn kein spezieller *Port* angegeben ist, wird der Kontakt über den Standardport von TELNET (23/tcp) hergestellt.

quit Schließt alle offenen TELNET-Sessions und beendet das Programm.

status Zeigt den aktuellen Status von **telnet** an. Der Netzwerkrechner, zu dem eine Verbindung besteht, und der aktuelle Modus werden ausgegeben.

send *Argument*

Sendet eine oder mehrere spezielle Zeichensequenzen zum entfernten Rechner. Die folgenden Argumente können angegeben werden:

? Gibt eine Kurzbeschreibung zum Befehl **send** aus.

abort Sendet die TELNET-Sequenz ABORT.

ao Sendet die TELNET-Sequenz AO, die beim entfernten System dazu führen sollte, daß der Inhalt des Ausgabepuffers sofort auf das Terminal übertragen wird.

ayt Sendet die TELNET-Sequenz AYT (Are You There).

brk Sendet die TELNET-Sequenz BRK (Break).

ec Sendet die TELNET-Sequenz EC (Erase Character), die dazu führt, daß beim entfernten System das zuletzt empfangene Zeichen gelöscht wird.

el Sendet die TELNET-Sequenz EL (Erase Line), die zum Löschen der zuletzt eingegebenen Zeile auf dem entfernten System führt.

eof Sendet die TELNET-Sequenz EOF (End Of File).

eor Sendet die TELNET-Sequenz EOR (End Of Record).

escape Sendet den aktuellen Wert für das TELNET-Fluchtzeichen (voreingestellt mit ^]).

ga Sendet die TELNET-Sequenz GA (Go Ahead).

getstatus Wenn der entfernte Rechner den TELNET-Befehl STATUS unterstützt, fordert **getstatus** die aktuelle Statusinfomation von der Gegenstelle an.

ip Sendet die TELNET-Sequenz IP (Interrupt Process), welche die Gegenstelle zur Unterbrechung des gerade laufenden Prozesses auffordert.

nop Sendet die TELNET-Sequenz NOP (No OPeration).

→

susp Sendet die TELNET-Sequenz SUSP (SUSPend process).

synch Sendet die TELNET-Sequenz SYNCH, die alle vorher eingegebenen (aber noch nicht gelesenen) Eingabedaten verwirft.

set *Argument Wert*

unset *Argument Wert*

Setzt eine von mehreren Telnet-Variablen auf einen angegebenen *Wert* oder auf WAHR. Der Spezialwert „off" (aus) schaltet die mit der Variablen verbundene Funktion aus. **unset** schaltet jede der angegebenen Funktionen ab. Die Werte der Variablen können mit Hilfe des **display**-Befehls angezeigt werden. Folgende Variablen können auf diesem Weg gesetzt werden:

? Gibt eine Liste der erlaubten **set**- und **unset**-Befehle aus.

ayt Wenn TELNET im **localchars**-Modus arbeitet, wird das angegebene Zeichen als alternatives AYT-Zeichen verwendet.

echo Das ist der Wert, der im Line-by-line-Modus zwischen lokalem Echo und Unterdrücken des Echos umschaltet (Voreinstellung ^E).

eof Wenn **telnet** im LINEMODE oder im alten Line-by-line-Modus arbeitet, bewirkt die Eingabe dieses Zeichens als erstes Zeichen einer Zeile, daß das Zeichen an das entfernte System geschickt wird.

erase Wenn **telnet** im **localchars**-Modus ist und im Character-at-a-time-Modus arbeitet, führt die Eingabe dieses Zeichens zum Senden der TELNET-Sequenz EC.

escape Dies ist das TELNET-Fluchtzeichen (Voreinstellung ^[), das die Umschaltung in den TELNET-Befehlsmodus bewirkt, wenn eine Verbindung zu einem entfernten System besteht.

flushoutput

Wenn **telnet** im **localchars**-Modus ist und das **flushoutput**-Zeichen eingegeben wird, wird die TELNET-Sequenz AO an das entfernte System gesendet.

forw1 Wenn **telnet** im **localchars**-Modus ist, wird dieses Zeichen als alternatives End-of-line-Zeichen verwendet.

forw2 Wenn **telnet** im **localchars**-Modus ist, wird dieses Zeichen als alternatives End-of-line-Zeichen verwendet.

interrupt Wenn **telnet** im **localchars**-Modus ist und das **interrupt**-Zeichen eingegeben wird, wird die TELNET-Sequenz IP an den entfernten Rechner gesendet.

kill Wenn **telnet** im **localchars**-Modus ist und im Character-at-a-time-Modus arbeitet, führt dieses Zeichen zum Senden der TELNET-Sequenz EL.

lnext Wenn **telnet** im LINEMODE oder im Line-by-line-Modus ist, wird dieses Zeichen als **lnext**-Zeichen für das Terminal verwendet.

quit Wenn **telnet** im **localchars**-Modus ist und das **quit**-Zeichen eingegeben wird, wird die TELNET-Sequenz BRK an den entfernten Rechner geschickt.

reprint Wenn **telnet** im LINEMODE oder Line-by-line-Modus ist, wird dieses Zeichen als das **reprint**-Zeichen für das Terminal verwendet.

start Wenn die TOGGLE-FLOW-CONTROL-Option eingeschaltet ist, wird dieses Zeichen als **start**-Zeichen für das Terminal verwendet.

stop Wenn die TOGGLE-FLOW-CONTROL-Option eingeschaltet ist wird dieses Zeichen als **stop**-Zeichen für das Terminal verwendet.

susp Wenn **telnet** im LINEMODE oder im **localchars**-Modus ist und das **suspend** Zeichen eingegeben wird, wird die TELNET-Sequenz SUSP an den entfernten Rechner gesendet.

tracefile Die Datei, in welche die Ausgabe von **netdata** geschrieben wird.

worderase Wenn **telnet** im LINEMODE oder im Line-by-line-Modus ist, wird dieses Zeichen als das **worderase**-Zeichen für das Terminal verwendet. Die Voreinstellung ist terminalspezifisch.

slc [*state*]

Setzt den Status der Sonderzeichen, wenn die TELNET LINEMODE-Option eingeschaltet ist.

? Gibt eine Kurzhilfe zum **slc**-Befehl aus.

check Überprüft die aktuelle Einstellung der Sonderzeichen. Wenn Abweichungen auftreten, werden die lokalen Einstellungen denen der entfernten Maschine angepaßt.

export Benutzt die lokalen Voreinstellungen für die Sonderzeichen.

import Benutzt die Einstellungen der entfernten Maschine für die Sonderzeichen.

environ [*Argumente* [. . .]]

Manipuliert die Variablen, die mittels der TELNET ENVIRON-Option weitergegeben werden können. Gültige Argumente für **environ** sind:

define *Variable Wert*

Setzt die *Variable* auf den *Wert*.

undefine *Variable*

Entfernt die *Variable* aus der Liste der Umgebungsvariablen.

export *Variable*

Markiert die zum entfernten Rechner zu exportierende *Variable*.

→

unexport *Variable*
: Markiert die *Variable* als nicht automatisch zu exportieren. Die Variable kann explizit von der Gegenstelle angefordert werden.

list
: Zeigt die aktuelle Variablenliste an.

send *Variable*
: Sendet die *Variable* an die Gegenstelle.

toggle *Argumente* [. . .]
: Schaltet verschiedene Flags zur Steuerung der TELNET-Reaktion auf Ereignisse. Die Flags können mit den Befehlen **set** und **unset** explizit auf WAHR oder FALSCH gesetzt werden. Die gültigen Argumente sind:

?
: Gibt eine Liste der erlaubten **toggle**-Befehle aus.

autoflush
: Wenn **autoflush** und **localchars** gleichzeitig WAHR sind und **ao**- oder **quit**-Zeichen erkannt werden, gibt **telnet** so lange keine Daten auf dem Benutzerterminal aus, bis das entfernte System die Verarbeitung dieser TELNET-Sequenzen bestätigt.

autosynch
: Wenn **autosynch** und **localchars** gleichzeitig WAHR sind und **intr**- oder **quit**-Zeichen eingegeben werden, wird an die resultierende TELNET-Sequenz automatisch eine SYNCH-Sequenz angehängt. Die Voreinstellung dieses Schalters ist FALSCH.

binary
: Schaltet die TELNET-Option BINARY für die Eingabe und Ausgabe ein bzw. aus.

inbinary
: Schaltet die TELNET-Option BINARY für die Eingabe ein bzw. aus.

outbinary
: Schaltet die TELNET-Option BINARY für die Ausgabe ein bzw. aus.

crlf
: Wenn dieser Schalter WAHR ist, werden Carriage Returns als Kombination CR-LF gesendet, anderenfalls werden sie unverändert weitergegeben (Voreinstellung ist FALSCH).

crmod
: Schaltet den Wagenrücklaufmodus. Die Voreinstellung ist FALSCH.

debug
: Schaltet den Fehlersuchmodus auf Socketebene. Voreinstellung ist FALSCH.

localchars
: Wenn dieser Wert WAHR ist, werden die Sonderzeichen **flush**, **interrupt**, **quit**, **erase** und **kill** lokal bearbeitet und in die passenden TELNET-Steuersequenzen umgeformt. Der Initialwert ist WAHR.

netdata
: Schaltet den Anzeigemodus für Netzwerkdaten. Voreinstellung ist FALSCH.

options
: Schaltet die Anzeige einiger interner, mit Optionen verbundener **telnet**-Protokoll-Prozeduren. Voreinstellung ist FALSCH.

prettydump
: Wenn **netdata** und **prettydump** beide WAHR sind, geschieht die Ausgabe der Netzwerkdaten in einer etwas lesbareren Form.

skiprc
: Unterdrückt die Bearbeitung der Datei *~/.telnetrc*. Die Voreinstellung ist FALSCH, die Datei wird also verarbeitet.

termdata
: Schaltet die hexadezimale Ausgabe von Terminaldaten. Die Voreinstellung ist FALSCH.

---

test *Ausdruck*
oder
[*Ausdruck*]

(Auch als eingebauter Befehl in den meisten Shells enthalten.)

Bewertet einen *Ausdruck* und liefert den Statuswert Null, wenn der Wert WAHR ist. Anderenfalls wird ein Statuswert ungleich Null zurückgegeben. In Shellskripts kann auch die alternative Form [*Ausdruck*] verwendet werden. Das Programm wird in der Regel für die Konstruktion der Bedingung für Programmverzweigungen in Shell-Programmen benutzt.

Dateitests

Die Syntax für die folgenden Optionen ist **test** *Option Datei*. Wenn die angegebene Datei nicht existiert, liefert das Programm den Wert FALSCH, anderenfalls wird der durch den Optionsschalter festgelegte Test auf die Datei angewendet, wie in der Beschreibung erklärt.

-b Ist die Datei ein blockorientiertes Gerät?

-c Ist die Datei ein zeichenorientiertes Gerät?

-d Ist die Datei ein Verzeichnis?

-e Existiert die Datei?

-f Ist die Datei nicht mehr als eine reguläre Datei?

-g Ist das Gruppen-ID-Bit bei der Datei gesetzt?

-k Ist das Sticky-Bit bei der Datei gesetzt?

-L Ist die Datei ein symbolischer Link?

-p Ist die Datei eine Named Pipe (FIFO)?

-r Ist die Datei für den Benutzer lesbar?

-s Ist die Datei nicht leer?

-S Ist die Datei ein Socket?

-t [*file-descriptor*]
: Gehört der *file-descriptor* (Voreinstellung ist Standardausgabe) zu einem Terminal?

-u Ist das Benutzer-ID-Bit bei der Datei gesetzt?

-w Ist die Datei für den Benutzer beschreibbar?

-x Ist die Datei ausführbar?

-O Gehört die Datei dem effektiven Eigentümer des laufenden Prozesses?

-G Gehört die Datei der effektiven Gruppe des laufenden Prozesses?

→

test
←

Dateivergleiche

Die Syntax für Dateivergleiche ist **test** *Datei1 Option Datei2.*

- **-nt** Ist die *Datei1* neuer als *Datei2*? Testet das Datum der letzten Änderung, nicht das der Erzeugung.
- **-ot** Ist die *Datei1* älter als *Datei2*? Testet das Datum der letzten Änderung, nicht das der Erzeugung.
- **-ef** Haben die Dateien identische Geräte- und Inode-Nummern?

Zeichenkettentests

Die Zeichenkettentests haben die Syntax **test** *Option Zeichenkette.*

- **-z** Ist die Zeichenkette leer?
- **-n** Ist die Zeichenkette mindestens ein Zeichen lang?
- = *Zeichenkette*
  Sind die beiden Zeichenketten gleich?
- != *Zeichenkette*
  Sind die beiden Zeichenketten verschieden?

Ausdrucktests

Beachten Sie, daß als Ausdruck jeder der bisher vorgestellten Tests erscheinen kann.

- ! *Ausdruck*
  Ist der Ausdruck falsch?
- *Ausdruck* -a *Ausdruck*
  Sind beide Ausdrücke wahr?
- *Ausdruck* -o *Ausdruck*
  Ist einer der Ausdrücke wahr?

Ganzzahltests

Die Syntax für Ganzzahltests ist **test** *Zahl1 Option Zahl2.* Sie können anstelle einer Zahl auch den Ausdruck -l *Zeichenkette* verwenden, der durch die Länge der *Zeichenkette* ersetzt wird.

- **-eq** Sind die beiden Zahlen gleich?
- **-ne** Sind die beiden Zahlen ungleich?
- **-lt** Ist *Zahl1* kleiner als *Zahl2*?
- **-le** Ist *Zahl1* kleiner als oder gleich *Zahl2*?
- **-gt** Ist *Zahl1* größer als *Zahl2*?
- **-ge** Ist *Zahl1* größer als oder gleich *Zahl2*?

tftp

tftp [*Host* [*Port*]]

Benutzerschnittstelle zum TFTP (Trivial File Transfer Protocol), das den Transfer von Dateien zwischen einem entfernten Rechner und der lokalen Maschine ermöglicht.

Befehle

Wenn das Programm gestartet wurde, erscheint ein Prompt:

```
tftp>
```

Das Programm versteht die folgenden Befehle:

? [*Befehl*. . .]
: Gibt Hilfsinformationen (zum *Befehl*) aus.

ascii Abkürzung für „mode ASCII"

binary Abkürzung für „mode binary"

connect *Hostname* [*Port*]
: Stellt die Verbindung zum angegebenen *Host* her und benutzt den optional angegebenen *Port*.

get *Dateiname*

get *Remotename Lokalname*

get *Datei1 Datei2 Datei3* . . .
: Holt eine Datei oder eine Gruppe von Dateien von dem entfernten System.

mode *Transfermodus*
: Stellt den *Transfermodus* ein. Der *Transfermodus* kann entweder ASCII oder binary sein, Voreinstellung ist ASCII.

put *Dateiname*

put *Lokalname Remotename*

put *Datei1 Datei2 Datei3* . . . *Verzeichnis*
: Sendet eine Datei oder eine Gruppe von Dateien an das entfernte System.

quit Beendet **tftp**.

rexmt *Timeout*
: Stellt den Timeout-Wert für die Paket-Retransmission in Sekunden ein.

status Gibt Statusinformationen zum Programm: ob eine Verbindung besteht, die aktuellen Timeout-Werte, den Trace-Modus.

timeout *Timeout*
: Setzt den Timeout-Wert für die gesamte Übertragung.

trace Schaltet Tracing ein bzw. aus.

verbose
: Schaltet den Verbose-Modus ein bzw. aus.

---

tload [*Optionen*] [*TTY*]

Stellt die mittlere Systemlast in einer einfachen ASCII-Grafik dar. Wenn ein *TTY* angegeben ist, wird die Ausgabe auf dieses Gerät gelenkt.

Optionen

-d Gibt die Verzögerung zwischen zwei Updates in Sekunden an.

-s Stellt die Skalierung ein (Anzahl der Zeilen für eine Einheit). Eine kleinere Zahl führt zu einer gröberen Skala.

---

top

Liefert regelmäßig aktualisierte Information über die am meisten CPU-intensiven Prozesse. Sehen Sie **ps**, um eine Erklärung der Feldbezeichnungen zu erhalten.

→

top
←

Optionen

-d
: Gibt die Verzögerung zwischen zwei Aktualisierungen an.

-i
: Unterdrückt die Anzeige von Idle-Prozessen.

-q
: Aktualisierung ohne Verzögerung. Unter Rootrechten läuft der Prozeß mit höchster Priorität.

-s
: Secure-Mode, schaltet einige (gefährliche) interaktive Befehle ab.

-S
: Kumulations-Modus. Es wird die gesamte CPU-Zeit jedes Prozesses angezeigt und die der toten Kinder hinzugezählt.

Interaktive Befehle

^L
: Schreibt den Bildschirminhalt neu.

k
: Erfragt eine Prozeß-ID und ein Signal und sendet das Signal zum Prozeß.

i
: Schaltet die Unterdrückung der Idle-Anzeige an bzw. aus.

n, #
: Erfragt die Anzahl der angezeigten Prozesse. Wird hier 0 angegeben, werden so viele Prozesse angezeigt, wie auf dem Bildschirm darstellbar sind (Voreinstellung).

q
: Beendet das Programm.

r
: Führt ein **renice** für einen Prozeß durch. Es werden PID und **renice**-Wert abgefragt. Im Secure-Modus abgeschaltet.

S
: Schaltet den kumulativen Modus ein bzw. aus (siehe Option -S).

s
: Ändert die Verzögerung zwischen zwei Aktualisierungen. Die neue Verzögerungszeit wird erfragt. Im Secure-Modus abgeschaltet.

---

tr

tr [*Optionen*] [*Zeichenkette1* [*Zeichenkette2*]]

Übersetzt Zeichen — kopiert Daten von der Standardeingabe zur Standardausgabe und ersetzt Zeichen aus *Zeichenkette1* in *Zeichenkette2* oder löscht Zeichen aus *Zeichenkette1*.

Optionen

-c, --**complement**
: Veranlaßt die Benutzung des Komplements von *Zeichenkette1* bezüglich ASCII 001-377.

-d, --**delete**
: Löscht alle Zeichen aus *Zeichenkette1* aus dem Eingabestrom.

-s, --**squeeze-repeats**
: Ignoriert wiederholt auftretende Zeichen in *Zeichenkette2*.

Sonderzeichen

\a	^G (Glocke)
\b	^H (Backspace)
\f	^L (Seitenvorschub)
\n	^J (Zeilenvorschub)
\r	^M (Wagenrücklauf)
\t	^I (Tab)

\v ^K (vertikaler Tabulator)

nnn Zeichen mit dem Oktalwert *nnn*.

\\\\ Das Zeichen „Backslash"

char1-char2
: Alle Zeichen im Bereich von *char1* bis *char2*. Wenn *char1* in der Sortierung nicht vor *char2* liegt, wird eine Fehlermeldung ausgegeben.

[*char***Zahl*]
: Die Klammern müssen mit eingegeben werden. Der Ausdruck wird erweitert zu der entsprechenden Anzahl des angegebenen Zeichens. Beispielsweise wird **[x*4]** zu **xxxx**.

[:*Klasse*]
: Die Klammern müssen mit eingegeben werden. Der Ausdruck wird erweitert zu allen Zeichen der angegebenen *Klasse*.

alnum	Buchstaben und Zahlen
alpha	Buchstaben
blank	Leerzeichen und Tabulatoren
cntrl	Kontrollzeichen
digit	Ziffern
graph	Druckbare Zeichen (ohne Space)
lower	Kleinbuchstaben
print	Druckbare Zeichen
punct	Satzzeichen
space	Whitespace (horizontal oder vertikal)
upper	Großbuchstaben
xdigit	Hexadezimalziffern

[=*char*=]
: Die Zeichenklasse, zu der das Zeichen *char* gehört. Nicht vollständig implementiert.

Beispiele

Wandelt in einer Datei alle Großbuchstaben in Kleinbuchstaben um:

```
cat file | tr '[A-Z]' '[a-z]'
```

Wandelt alle Leerzeichen in Zeilenvorschübe (ASCII-Code 012) um:

```
tr ' ' '\012' < file
```

Entfernt Leerzeilen aus der Datei **file** und schreibt in die neue Datei **new.file** (oder entfernt wiederholte Tabulatoren durch **011**):

```
cat file | tr -s "" "\012" > new.file
```

Entfernt Doppelpunkte aus der Datei **file** und schreibt in die neue Datei **new.file**:

```
tr -d : < file > new.file
```

true	**true** Dieses Programm tut nichts, liefert jedoch einen erfolgreichen Rückkehrstatus (0). Siehe auch **false**.
ul	**ul** [*Optionen*] [*Datei*] Wandelt Sequenzen aus Zeichen-Rückschritt-Unterstrich in das Attribut „Unterstrichen" bei der Terminaldarstellung von Zeichen. Die korrekte Umwandlung ist abhängig vom Terminaltyp und wird durch die Umgebungsvariable TERM ermittelt. Einige Terminals sind nicht in der Lage, unterstrichene Zeichen darzustellen. *Optionen* -i Unterstrichene Zeichen werden durch ein – auf einer separaten Zeile dargestellt. -t *Terminaltyp* Legt den *Terminaltyp* fest.
umount	**umount** *Argument* Setzt Dateisysteme ab. Details finden Sie bei der Beschreibung von **mount**.
unexpand	**unexpand** [*Optionen*] [*Datei*] Wandelt Zeichenketten von führenden Leerzeichen, die aus mindestens zwei Spaces oder Tabs bestehen, zu Tabs um. Wenn keine *Datei* oder ein – als Dateiname angegeben ist, wird von der Standardeingabe gelesen. *Optionen* -, -t, --tabs *tab-stops* *tab-stops* ist eine durch Kommata getrennte Liste von Ganzzahlen, welche die Plazierung der Tabulatoren festlegt. Wenn genau ein Ganzzahlwert angegeben ist, werden die Tabulatoren in gleichmäßigem Abstand dieser Schrittweite angeordnet. Voreingestellt ist eine Tabulatorweite von acht Zeichen. Die Verwendung von -t impliziert -a. Bei Verwendung von -t und --tabs darf die Liste auch durch Leerzeichen anstelle von Kommata separiert werden. -a, --all Wandelt alle, nicht nur die führenden Leezeichen um.
uniq	**uniq** [*Optionen*] [*Datei1* [*Datei2*]] Entfernt doppelte aufeinanderfolgende Zeilen aus einer sortierten *Datei1* und schreibt die einzelnen Zeilen in die *Datei2* (oder in die Standardausgabe). Das Programm wird häufig als Filter benutzt. Benutzen Sie nur eine der Optionen -c, -d oder -u. Siehe auch **comm** und **sort**.

Optionen

-*n*, -f *n*, --skip-fields=*n*
: Ignoriert die ersten *n* Felder einer Zeile. Felder sind durch Leerzeichen voneinander getrennt.

+*n*, -s *n*, --skip-chars=*n*
: Ignoriert die ersten *n* Zeichen eines Feldes.

-c, --count
: Gibt jede Zeile einmal aus und stellt jeder Zeile die Anzahl ihres Auftretens voran.

-d, --repeated
: Gibt nur die doppelt vorkommenden Zeilen aus.

-u, --unique
: Gibt nur die Zeilen aus, die nicht doppelt vorkommen.

-w *n*, --check-chars=*n*
: Es werden nur die ersten *n* Zeichen jeder Zeile verglichen (beginnend mit dem ersten nicht ignorierten Feld bzw. Zeichen).

Beispiele

Schreibt eine Kopie jeder Zeile der Datei **list** in die Ausgabedatei **list.new**:

```
uniq list list.new
```

Zeigt, welche Namen mehr als einmal auftreten:

```
sort names | uniq -d
```

Zeigt, welche Zeilen exakt dreifach auftreten:

```
sort names | uniq -c | grep "3 "
```

uptime

Zeigt die aktuelle Zeit, die Zeit seit dem letzten Reboot, die Anzahl der gerade eigeloggten Benutzer und die durchschnittliche Systemlast an. Die gleiche Ausgabe erscheint in der ersten Zeile des Programms **w**.

users [*Datei*]

Gibt eine durch Leerzeichen separierte Liste aller Login-Sessions des lokalen Rechners aus. Der gleiche Benutzer kann mehrfach auftreten. Das Programm bezieht seine Daten aus der angegebenen *Datei* oder aus */etc/utmp*.

vi [*Optionen*] [*Dateien*]

Ein bildschirmorientierter Texteditor auf der Basis von **ex**. Weitere Informationen über **vi** finden Sie in Kapitel 7.

vrfy

vrfy [*Optionen*] *Adresse* [*Host*]

Fragt bei einem entfernten Rechner an, um die Korrektheit einer Email-Adresse zu verifzieren. Wenn ein *Host* angegeben ist, wird dieser Rechner direkt kontaktiert, anderenfalls kann die Anfrage durch das Mail-System gehen. Es werden Informationen über die Adresse und, wenn verfügbar, über den Benutzer ausgegeben. Mit den Optionen **-f** und **-p** muß die Adresse nicht auf der Kommandozeile angegeben werden.

Optionen

-a
: Führt Anfragen bei allen Mail-Exchange-Rechnern durch, nicht nur beim primären.

-c *Sekunden*
: Setzt den Timeout für den Verbindungsaufbau auf die angegebene Anzahl *Sekunden* (Voreinstellung ist 6).

-d
: Schaltet die Ausgabe zur Fehlersuche ein.

-dd
: Schaltet weitere Ausgabe zur Fehlersuche hinzu.

-e
: Veranlaßt die Verwendung des Befehls EXPN anstelle von VRFY, wenn eine Verbindung zum Sendmail-Port der Gegenstelle besteht.

-f *Datei*
: Liest die Liste der zu verifizierenden Email-Adressen aus der angegebenen *Datei*.

-l
: Schaltet die lokale Fehlerbehandlung ein.

-p *Domain*
: Prüft, ob die Mail-Exchanger der *Domain* SMTP-Requests akzeptieren.

-s
: Entfernt Kommentare von der Hostadresse.

-n
: Veranlaßt die Benutzung von RCPT anstelle von VRFY, wenn eine Verbindung zum Sendmail-Port der Gegenstelle besteht. Rekursionen werden unterdrückt.

-t *Sekunden*
: Setzt den Timeout für Leseversuche auf die angegebene Anzahl *Sekunden* (Voreinstellung ist 60).

-v
: Verbose.

-vv
: Very verbose.

-vvv
: Amazingly verbose.

-L *Levels*
: Verifiziert rekursiv die angegebene Anzahl von *Levels* der Adresse. **-L** erkennt Schleifen und lokale Benutzer und beendet bei deren Auftreten die Prüfung.

-R
: Entspricht **-L 17**. 17 Stationen sind die Obergrenze bei realen Mailings. Diese Option impliziert **-s**.

w

w [*Optionen*] [*Benutzer*]

Gibt eine Zusammenfassung der Systemauslastung, der aktuell eingeloggten Benutzer und deren Prozesse aus. **w** ist eine Kombination von **uptime**, **who** und **ps -a**. Die Ausgabe wird auf einen Benutzer beschränkt, indem dessen Name auf der Kommandozeile übergeben wird.

Optionen

-h Unterdrückt die Ausgabe von **uptime** und die Kopfzeile von **ps**.

-i Die Ausgabe wird nach Leerlaufzeit sortiert.

wc [*Optionen*] [*Dateien*]

Gibt jeweils die Anzahl für Zeichen, Wörter und Zeilen der angegebenen *Dateien* aus. Wenn keine *Datei* angegeben ist, wird von der Standardeingabe gelesen. Weitere Beispiele finden Sie bei den Programmbeschreibungen für **ls** und **sort**.

Optionen

-c, -bytes, --chars
: Es wird nur die Zeichenzahl ausgegeben.

-l, --lines
: Es wird nur die Zeilenzahl ausgegeben.

-w, --words
: Es wird nur die Zahl der Wörter ausgegeben.

Beispiele

Zählt die Anzahl der eingeloggten Benutzer:

```
who | wc -l
```

Zählt die Wörter in drei Essay-Dateien:

```
wc -w essay.[123]
```

Zählt die Zeilen in der Datei, deren Name in der Variablen **$file** enthalten ist (ohne Ausgabe des Dateinamens):

```
wc -l < $file
```


whatis [*Optionen*] *Schlüsselwort*

Durchsucht die Kurzbeschreibungen der Manualpages nach dem angegebenen *Schlüsselwort*. Es wird die Kurzbeschreibung für jede Fundstelle ausgegeben.

Optionen

-d, --debug
: Schaltet die Ausgabe zur Fehlersuche hinzu.

-m *Systeme*, --systems=*Systeme*
: Durchsucht die Manualpages eines anderen Betriebssystems.

-r, --regex
: Behandelt das *Schlüsselwort* als regulären Ausdruck.

-w, --wildcard
: Erlaubt im *Schlüsselwort* die Benutzung von Jokerzeichen (wie bei der Shell).

-M *Pfad*, --manpath=*Pfad*
: Durchsucht den angegebenen *Pfad* nach Manualpages.

who

who [*Optionen*] [*Datei*]
who am i

Zeigt Informationen über den aktuellen Status des Systems an. Ohne Angabe von Optionen werden eine Liste der aktuell eingeloggten Benutzer, deren Terminal, die Loginzeit und der Name des Rechners, von dem aus sie sich eingeloggt haben, angezeigt. Eine *Datei* kann als Ersatz für */var/run/utmp* angegeben werden, aus der die Systeminformationen gelesen werden.

Optionen

am i Gibt den Namen des Benutzers aus.

-m Entspricht **who am i**.

-i, -u, --idle
Gibt zusätzlich noch die Leerlaufzeit aus. Eine Leerlaufzeit von . zeigt eine Benutzeraktivität innerhalb der letzten Minute an, das Wort **old** hingegen zeigt Inaktivität von mehr als einem Tag an.

-q, --count
Es werden nur die Benutzernamen und die Gesamtzahl der eingeloggten Benutzer angezeigt.

-w, -T, --mesg, --message, --writable
Zeigt den Message-Status der Benutzer an.

+ mesg y (Messages können mit **write** geschrieben werden)

\- mesg n (Messages können nicht geschrieben werden)

? unbekannt

-H, ––heading
Veranlaßt die Ausgabe einer Kopfzeile.

Beispiel

Diese Beispielausgabe wurde am 17. April um 8 Uhr morgens erzeugt:

```
who -uH
NAME    LINE   TIME          IDLE   PID  COMMENTS
Earvin  ttyp3  Apr 16 08:14  16:25  2240
Larry   ttyp0  Apr 17 07:33    .   15182
```

Benutzer Earvin ist inaktiv seit dem Nachmittag des Vortages. Es sieht also so aus, als sei er noch nicht bei der Arbeit, er hat sich aber nicht ausgeloggt. Larry's Terminal wird gerade benutzt.

whoami

whoami

Gibt die Identifikationsnummer des Benutzers aus. Entspricht **who am i** und **id -un**.

write

write *Benutzer* [*TTY*]
Nachricht

Startet eine Erwiderung oder einen interaktiven Dialog mit dem *Benutzer*.

Eine **write**-Session wird durch *EOF* beendet. Wenn der Benutzer auf mehr als einem Terminal eingeloggt ist, geben Sie die Nummer des gewünschten Terminals an. Siehe auch **talk**.

write

xargs [*Optionen*] [*Kommando*]

xargs

Führt das angegebene *Kommando* (mit beliebigen Anfangsargumenten) aus, liest aber die verbleibenden Argumente von der Standardeingabe, statt sie direkt anzugeben. **xargs** übergibt diese Argumente in mehreren Paketen an den *Befehl* und erlaubt damit dem *Befehl*, mehr Argumente zu behandeln als normal. Die Argumente sind typischerweise lange Listen mit Dateinamen (zum Beispiel aus **ls** oder **find**), die via Pipe an **xargs** übergeben werden.

Optionen

-0, --null
: Erwartet, daß Dateinamen durch NULL-Bytes terminiert sind. Quotes und Backslashes werden nicht besonders behandelt.

-e[*Zeichenkette*], **--eof**[=*Zeichenkette*]
: Setzt *EOF* auf _ oder auf die angegebene *Zeichenkette*.

-i[*Zeichenkette*], **--replace**[=*Zeichenkette*]
: Ersetzt das Auftreten von { } oder der optionalen *Zeichenkette* in der Befehlszeile durch die von der Standardeingabe gelesenen Namen. Leerzeichen ohne Anführungszeichen werden nicht als Trenner zwischen Argumenten behandelt. Diese Option impliziert **-x** und **-l 1**.

-l[*Zeilen*], **--max-lines**[=*Zeilen*]
: Begrenzt die Anzahl der erlaubten Eingabezeilen für ein Kommando. Ohne Angabe von *Zeilen* wird die Zeilenzahl auf 1 begrenzt. Diese Option impliziert **-x**.

-n *args*, **--max-args**=*args*
: Begrenzt die Anzahl der Argumente für ein Kommando. Diese Option kann durch **-s** verdrängt werden.

-p, --interactive
: Fragt bei jeder *Befehl*sausführung nach der Bestätigung durch **y**. Diese Option impliziert **-t**.

-r, --no-run-if-empty
: Unterdrückt die Ausführung eines Kommandos, das nur aus Leerzeichen besteht.

-s *max*, **--max-chars**=*max*
: Erlaubt nicht mehr als *max* Zeichen pro Kommandozeile.

-t, --verbose
: Verbose-Modus.

-x, --exit
: Führt zum Abbruch des Programms, wenn die maximale Länge der Kommandozeile (wie durch **-s** festgelegt) überschritten wird.

-P *max*, **--max-procs**=*max*
: Begrenzt die Anzahl der gleichzeitig laufenden Prozesse auf die Zahl *max*. Voreinstellung ist 1. Ein Maximum von 0 hebt die programminterne Begrenzung der Prozeßzahl auf.

→

xargs
←

Beispiele

Durchsucht (**grep**) alle Dateien des Systems nach dem *Muster*:

```
find / -print | xargs grep Muster > out &
```

Führt **diff** mit Dateipaaren aus (z.B. **f1.a** und **f1.b**, **f2.a** und **f2.b**):

```
echo $* | xargs -n2 diff
```

Die vorhergehende Zeile wird als Shell-Prozedur aufgerufen, mit den Dateinamen als Argumenten.

Zeigt die *Datei* mit einem Wort pro Zeile an (entspricht **deroff -w**):

```
cat Datei | xargs -n1
```

Verschiebt Dateien aus **olddir** ins **newdir** und zeigt jeden Befehl an:

```
ls olddir | xargs -i -t mv olddir/{} newdir/{}
```

ypchfn

ypchfn [*Optionen*] [*Benutzername*]

NFS-/NIS-Programm. Ändert die in */etc/passwd* gespeicherte Information, die vom Finger-Dämon angezeigt wird. Ohne *Optionen* fragt **ypchfn** die Änderungen interaktiv ab. Um das gesamte Informationsfeld zu löschen, geben Sie das Schlüsselwort **none** ein. Siehe auch **yppasswd** und **ypchsh**.

Optionen

-f Das Programm arbeitet wie **chfn** (Default).

-l Das Programm arbeitet wie **ypchsh**.

-p Das Programm arbeitet wie **yppasswd**.

ypchsh

ypchsh [*Optionen*] [*Benutzername*]

NFS-/NIS-Programm. Ändert die Loginshell und gibt diese Information über NIS weiter. Gibt eine Warnung aus, wenn die *shell* nicht in */etc/shells* eingetragen ist. Siehe auch **yppasswd** und **ypchfn**.

Optionen

-f Das Programm arbeitet wie **chfn**.

-l Das Programm arbeitet wie **ypchsh** (Default).

-p Das Programm arbeitet wie **yppasswd**.

yppasswd

yppasswd [*Optionen*] [*Benutzername*]

NFS-/NIS-Programm. Erzeugt oder ändert das Paßwort für den angegebenen *Benutzer* und gibt das neue Paßwort über NIS weiter. Nur der Eigentümer oder der Superuser darf ein Paßwort ändern. Der Eigentümer muß keinen *Benutzernamen* angeben. Siehe auch **ypchfn** und **ypchsh**.

Optionen

- **-f** Das Programm arbeitet wie **chfn**.
- **-l** Das Programm arbeitet wie **ypchsh**
- **-p** Das Programm arbeitet wie **yppasswd** (Default).

zcat [*Datei*]

Liest eine oder mehrere *Dateien*, die zuvor mit **gzip** oder **compress** komprimiert worden sind, und schreibt sie in die Standardausgabe. Liest von der Standardeingabe, wenn keine *Datei* oder - als Argument angegeben wurde. Das Programm wird in diesem Fall durch ein *EOF* beendet.

zcmp [*Optionen*] *Dateien*

Liest komprimierte Dateien und übergibt den unkomprimierten Inhalt zusammen mit allen Kommandozeilenoptionen an **cmp**. Wenn keine zweite Datei zum Vergleichen angegeben wurde, wird nach einer Datei mit dem Namen der ersten Datei und der zusätzlichen Endung **.gz** gesucht.

zdiff [*Optionen*] *Dateien*

Liest komprimierte Dateien und übergibt den unkomprimierten Inhalt zusammen mit allen Kommandozeilenoptionen an **diff**. Wenn keine zweite Datei zum Vergleichen angegeben wurde, wird nach einer Datei mit dem Namen der ersten Datei und der zusätzlichen Endung **.gz** gesucht.

zforce *Dateien*

Benennt alle mit **gzip** komprimierten Dateien um in *Dateiname*.gz, falls sie nicht bereits die Endung **.gz** haben.

zgrep [*Optionen*] [*Dateien*]

Liest komprimierte *Dateien* und übergibt den unkomprimierten Inhalt zusammen mit allen Kommandozeilenoptionen an **grep**. Wenn keine Datei angegeben wurde, liest das Programm von der Standardeingabe und versucht, diese Daten zu dekomprimieren. Das Programm kann auch unter den Namen **zegrep** oder **zfgrep** aufgerufen werden und benutzt in diesen Fällen **egrep** bzw. **fgrep** zum Durchsuchen des Dateiinhalts.

zmore [*Dateien*]

Ähnlich wie **more**. Gibt den unkomprimierten Inhalt einer Datei seitenweise aus.

Kommandos

Leerzeichen
: Gibt die nächste Bildschirmseite aus.

→

zmore ←

i[*Leerzeichen*]
: Gibt die durch die Zahl *i* angegebene Anzahl Zeilen aus.

^D
: Blättert die Anzeige um eine halbe Bildschirmseite (elf Zeilen) weiter. Durch eine Zahl vor dem Kommando kann die Anzahl der vorgeschobenen Zeilen eingestellt werden. Die Einstellung bleibt für die folgenden Kommandos dieses Typs erhalten.

d
: Das gleiche wie ^D.

*i*z
: Gibt die nächsten *i* Zeilen aus, wie mit Leerzeichen, jedoch wird die angegebene Zahl zur neuen Schrittweite für alle Kommandos dieses Typs.

*i*s
: Überspringt *i* Zeilen und gibt die darauffolgenden Zeilen aus.

*i*f
: Überspringt *i* Bildschirmseiten und gibt die darauffolgenden Zeilen aus.

q, Q
: Beendet die Anzeige der aktuellen Datei und beginnt mit der nächsten Datei oder beendet **zmore**.

e
: Exit.

s
: Überspringt die nächste Datei und zeigt die übernächste an.

=
: Zeigt die Zeilennummer der letzten Bildschirmzeile an.

i/*Ausdruck*
: Sucht vorwärts in allen Dateien nach dem *i*ten Vorkommen des (regulären) *Ausdrucks*. Es werden die Zeile mit der Fundstelle und die beiden vorhergehenden Zeilen angezeigt.

*i*n
: Sucht vorwärts nach dem *i*ten Vorkommen des zuletzt verwendeten regulären Ausdrucks.

!*Kommando*
: Führt das *Kommando* in einer Shell aus. Ein weiteres ! im Kommando wird im C-Shell-Stil durch das vorhergehende Shell-Kommando ersetzt.

:q, :Q
: Springt sofort zur nächsten Datei, genau wie **q**.

.
: Wiederholt das vorhergehende Kommando.

---

znew

znew [*Optionen*] [*Dateien*]

Dekomprimiert .Z-Dateien und rekomprimiert sie im .gz- Format.

Optionen

-9
: Optimale (und langsamste) Kompressionsmethode.

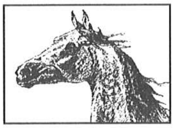
-f
: Erzwingt die Rekomprimierung, selbst wenn eine gleichnamige Datei mit der Endung **.gz** bereits existiert.

-t
: Testet die erzeugten **.gz**-Dateien, bevor die .Z- Originaldateien gelöscht werden.

-v
: Verbose-Modus.

-K
: Wenn die Originaldatei kleiner als die **.gz**-Datei ist, bleibt sie erhalten.

-P
: Veranlaßt die Benutzung einer Pipeline zwischen De- und Rekomprimierung zum Sparen von Plattenplatz.



KAPITEL 2

Die UNIX Shells: Ein Überblick

Die Shells sind Programme, die eine Vermittlerrolle zwischen Ihnen und dem Betriebssystem übernehmen. In ihrer Eigenschaft als Befehlsinterpreter sollten sie für Sie die meiste Zeit unsichtbar sein. Die Shells können auch für einfache Programme verwendet werden.

Dieser Abschnitt beschreibt drei Shells, die üblicherweise auf Linux-Systemen verwendet werden — die Bourne Again Shell (**bash**), die C Shell (**csh**) und die erweiterte Version der **csh**, die **tcsh** —, und faßt die wesentlichen Unterschiede zwischen ihnen zusammen. Die Kapitel 3, *bash: Die Bourne Again Shell* und 4, *csh und tcsh* enthalten Details zu den einzelnen Shells.

Wir besprechen hier die folgenden Themen:

- Sinn und Zweck einer Shell
- Arten von Shells
- Gemeinsamkeiten
- Unterschiede

Sinn und Zweck einer Shell

Es gibt drei Hauptanwendungen für eine Shell:

- Interaktive Benutzung
- Anpassung der eigenen Arbeitsumgebung
- Programmieren

Interaktive Shell-Benutzung

Wenn die Shell interaktiv benutzt wird, wartet sie auf die Eingabe Ihrer Befehle, verarbeitet sie (um spezielle Zeichen wie Joker behandeln zu können) und führt sie aus. Shells stellen

außerdem einige Befehle, die sogenannten eingebauten Befehle, bereit, die die Linux-Befehle ergänzen.

Anpassung der eigenen Arbeitsumgebung

Eine Linux-Shell definiert Variablen, z.B. die Lage Ihres Heimatverzeichnisses und Ihrer Mail-Datei, um das Verhalten während der Arbeit zu bestimmen. Einige Variablen werden bereits vom System gesetzt, andere können Sie in Startdateien definieren, die Ihre Shell einliest, wenn Sie sich beim System anmelden. Solche Startdateien können auch Linux- oder Shell-Befehle enthalten, die unmittelbar nach dem Anmelden ausgeführt werden.

Programmieren

Eine Reihe von einzelnen Befehlen (egal, ob Shell- oder andere verfügbare Linux-Befehle) können zu einem Programm, einem sogenannten Shell-Skript, zusammengefaßt werden. Die Stapel-Dateien von MS-DOS weisen ein ähnliches Konzept auf. Die **bash** ist eine mächtige Shell zur Programmierung, während das Schreiben von Programmen für die **csh** als gesundheitsgefährdend gilt.

Arten von Shells

Es gibt viele verschiedene Shells unter Linux. Diese Buch beschreibt die drei beliebtesten:

- Die Bourne Again Shell (**bash**), die auf der Bourne Shell basiert und unter Linux die Standard-Shell darstellt.
- Die C Shell (**csh**), die eine C-ähnliche Syntax verwendet und viele Bequemlichkeiten bietet.
- **tcsh**, eine Erweiterung der **csh**, die zusätzliche interessante Merkmale aufweist.

Die meisten Systeme stellen mehr als eine Shell zur Verfügung. Häufig wird auch eine Shell zum Schreiben von Shell-Skripten und eine andere für interaktive Benutzung verwendet.

Wenn Sie sich beim System anmelden, ermittelt das System die zu verwendende Shell, indem es Ihren Eintrag in der Datei */etc/passwd* liest. Das letzte Feld jedes Eintrags gibt ein Programm an, das standardmäßig als Shell gestartet werden soll. Ein Beispiel:

Wenn der Programmname folgender ist:	ist Ihre Shell die folgende:
/bin/sh	Bourne Again Shell
/bin/bash	Bourne Again Shell
/bin/csh	C Shell
/bin/tcsh	**tcsh**

Sie können zu einer anderen Shell wechseln, indem Sie den Programmnamen in die Kommandozeile eingeben. Um beispielsweise von der **bash** zur **tcsh** zu wechseln, geben Sie ein:

```
$ exec tcsh
```

Gemeinsamkeiten

Die untenstehende Tabelle ist eine Auswahl von Merkmalen, die **bash**, **csh** und **tcsh** gemeinsam haben. Bitte beachten Sie, daß die **tcsh** eine erweiterte Version der **csh** ist; daher enthält die **tcsh** alle Merkmale der **csh** sowie einige zusätzliche.

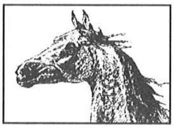
Symbol/Befehl	Bedeutung/Aktion
>	Ausgabe umleiten.
>>	Ausgabe an Datei anhängen.
<	Eingabe umleiten.
<<	"Here" Dokument (Eingabe umleiten).
|	Ausgabe weiterleiten.
&	Prozeß im Hintergrund starten.
;	Mehrere Befehle auf einer Zeile.
*	Paßt auf beliebig viele Zeichen in Dateinamen.
?	Paßt auf genau ein beliebiges Zeichen in Dateinamen.
!*n*	Wiederhole Befehl Nr. *n*.
[ ]	Paßt auf eines der Zeichen in den eckigen Klammern.
( )	Sub-Shell ausführen.
` `	Ausgabe des Kommandos zwischen den rückwärtigen Anführungsstrichen hier einfügen.
" "	Teilweises Quoten. (Expansion von Variablen und Befehlen erlaubt)
' '	Volles Quoten (keine Expansion erlaubt).
\	Nachfolgendes Zeichen quoten.
$*var*	Benutze den Wert der Variablen.
$$	Prozeßnummer.
$0	Befehlsname.
$*n*	*n*-tes Argument ($0<n\leq9$).
$*	Alle Argumente als einfaches Wort.
#	Anfang eines Kommentars.
bg	Ausführung im Hintergrund.
break	Schleifenbefehle abbrechen.
cd	Verzeichniswechsel.
continue	Schleife im Programm fortsetzen.
echo	Argumente ausgeben.
eval	Argumente auswerten.
exec	Anderes Programm anstelle der Shell ausführen.
fg	Ausführung im Vordergrund.
jobs	Aktive Jobs anzeigen.
kill	Laufende Jobs beenden.
newgrp	Gruppe wechseln.
shift	Über die Position bestimmte Parameter weiterschieben.
stop	Hintergrundjob anhalten.

Symbol/Befehl	Bedeutung/Aktion
suspend	Vordergrund anhalten.
time	Zeit zur Befehlsausführung messen.
umask	Maske für die Zugriffsrechte bei neu angelegten Dateien setzen oder anzeigen.
unset	Definition einer Variablen oder Funktion löschen.
wait	Beendigung eines Hintergrundjobs abwarten.

Unterschiede

Die nachstehende Tabelle zeigt eine Auswahl von Merkmalen, die in den drei Shells unterschiedlich ausgeprägt sind.

Bedeutung/Aktion	bash	csh	tcsh
Standard-Eingabeaufforderung.	$	%	%
Ausgabeumleitung erzwingen.	>|	>!	>!
Anhängen erzwingen.		>>!	>>!
Variablenzuweisung.	*Var=Wert*	set *Var=Wert*	set *Var=Wert*.
Umgebungsvariable setzen.	export *Var*	setenv *Var Wert*	setenv *Var Wert*
Anzahl der Argumente.	$#	$#argv	$#
Rückgabewert des letzten Befehls.	$?	$status	$?
Befehle in *Datei* ausführen.	. *Datei*	source *Datei*	source *Datei*
Schleife beenden.	done	end	end
case oder **switch** beenden.	esac	endsw	endsw
Schleife über Variablen.	for/do	foreach	foreach
Muster eines **if**-Befehls.	if [$i -eq 5]	if ($i==5)	if ($i==5)
Ende eines **if**-Befehls.	fi	endif	endif
Ressourcenbeschränkungen einstellen.	ulimit	limit	limit
Vom Terminal lesen.	read	$<	$<
Variable auf Nur-Lesen setzen.	readonly		set -r
Dateiabfrageoperator: prüft auf von Null verschiedene Größe.		-s	
Aktuelles Wort vervollständigen.	TAB		TAB
Interrupts ignorieren.	trap 2	onintr	onintr
until-Schleife anfangen.	until/do	until	until
while-Schleife beenden.	while/do	while	while



KAPITEL 3

bash: Die Bourne Again Shell

In diesem Kapitel werden die folgenden Themen besprochen:

- Funktionsübersicht
- Aufruf der Shell
- Syntax
- Variablen
- Arithmetische Ausdrücke
- Befehls-History
- Eingebaute Befehle
- Jobverwaltung

Funktionsübersicht

bash ist die Standard-Shell und bietet die folgenden Möglichkeiten:

- Umleitung von Ein- und Ausgabe
- Platzhalter-Zeichen (Metazeichen), um Dateinamen abkürzen zu können
- Shell-Variablen zur Einstellung der Arbeitsumgebung
- Mächtige eingebaute Programmiersprache
- Editieren der Kommandozeile (mit Editierbefehlen wie im **vi** oder im Emacs)
- Rückgriff auf früher eingegebene Befehle (Befehls-History)
- Ganzzahlrechenfunktionen
- Arithmetische Ausdrücke
- Abkürzen von Befehlen (aliasing)
- Jobverwaltung

- Eingebaute Programmier-Funktionen
- Kontrollstrukturen
- Abspeichern und Abrufen von Verzeichnissen (mit **pushd** und **popd**)
- Expansion von Ausdrücken mit geschweiften Klammern und Tilden
- Tastaturbindungen

Aufruf der Shell

Der Befehlsinterpreter der **bash** wird folgendermaßen aufgerufen:

bash [*Optionen*] [*Argumente*]

Die **bash** kann Befehle ausführen, die von einem Terminal (wenn **-i** angegeben ist), aus einer Datei (wenn das erste *Argument* ein ausführbares Skript ist) oder von der Standard-Eingabe (wenn keine Argumente oder **-s** angegeben sind) kommen.

Optionen

-, --
: Alle folgenden Zeichenketten werden als Argumente und nicht als Optionen behandelt.

-c *Str*
: Die in der Zeichenkette *Str* enthaltenen Befehle werden ausgeführt.

-i Eine interaktive Shell (mit Eingabeaufforderung) wird gestartet.

-login
: Die Shell verhält sich wie eine Login-Shell.

-nobraceexpansion
: Es findet keine Expansion von Ausdrücken in geschweiften Klammern statt.

-nolineediting
: Nur bei interaktiven Shells. Die Kommandozeile wird nicht mit Hilfe der GNU-readline-Bibliothek eingelesen. Kommandozeilen-Editierung ist kaum möglich ist.

-noprofile
: Die Dateien */etc/profile*, *~/.bash_profile*, *~/.bash_login*, und *~/.profile* werden beim Start nicht ausgeführt.

-norc
: Die Datei *~/.bashrc* wird beim Start nicht ausgeführt.

-p
: Die Shell wird als privilegierter Benutzer gestartet; die Datei *$HOME/.profile* wird nicht ausgeführt.

-posix
: POSIX-konformes Verhalten erzwingen.

-quiet
: Default. Es wird keine Start-Information ausgegeben.

-rcfile *Datei*
: Statt *.bashrc* wird beim Starten *Datei* verwendet.

-s Befehle werden von der Standard-Eingabe gelesen; die Ausgabe von eingebauten Befehlen wird an den Dateideskriptor 1, alle anderen Ausgaben werden an den Dateideskriptor 2 geschickt.

Alle weiteren Optionen der **bash** sind beim eingebauten **set**-Befehl aufgeführt.

Argumente

Die Argumente werden in der Reihenfolge ihres Auftretens den Positions-Parametern **$1**, **$2** usw. zugewiesen. Wenn das erste Argument ein ausführbares Skript ist, werden die Befehle daraus gelesen und ausgeführt und die verbleibenden Argumente entsprechend zugewiesen.

Syntax

Dieser Abschnitt beschreibt alle Symbole, die von der **bash** gesondert behandelt werden. Sie finden hier folgende Themen:

- Besondere Dateien
- Platzhalterzeichen in Dateinamen
- Editieren der Kommandozeile
- Quoten
- Befehlsausdrücke
- Umleitungsausdrücke
- Koprozesse

bash

Besondere Dateien

/etc/profile	Wird beim Anmelden automatisch ausgeführt.
$HOME/.bash_profile	Wird beim Anmelden automatisch ausgeführt.
$HOME/.bashrc	Wird beim Starten der Shell automatisch ausgeführt.
$HOME/.bash_logout	Wird beim Abmelden automatisch ausgeführt.
$HOME/.bash_history	Enthält die während der letzten Sitzung eingegebenen Befehle.
/etc/passwd	Enthält die Heimatverzeichnisse für ~*Benutzername*-Abkürzungen.

Platzhalterzeichen in Dateinamen

`*`	Paßt auf jede Zeichenkette mit null oder mehr Zeichen. (Paßt auf jede Zeichenkette.)
`?`	Paßt auf ein einzelnes Zeichen.
`[abc...]`	Paßt auf ein beliebiges der in den eckigen Klammern angegebenen Zeichen. Mit einem Bindestrich können Bereiche angegeben werden (z.B. a-z, A-Z, 0-9).
`[!abc...]`	Paßt auf jedes Zeichen, das *nicht* zwischen den eckigen Klammern angegeben ist.
`~Name`	Heimatverzeichnis des Benutzers *Name*
`~+`	Aktuelles Arbeitsverzeichnis (PWD)
`~-`	Vorheriges Arbeitsverzeichnis (OLDPWD)

Eine Folge von durch | getrennten Mustern ergibt wieder ein Muster. Wenn eines der Muster paßt, gilt der ganze Ausdruck als passend. Diese erweiterte Syntax ähnelt der von **egrep** und **awk**.

Beispiele

`$ ls neu*`	*Zeige* `neu` *and* `neu.1`
`$ cat kap?`	*Paßt auf* `kap9` *, aber nicht* `kap10`
`$ vi [D-R]*`	*Paßt auf alle Dateien, die mit einem großen Buchstaben beginnen, der zwischen D und R (jeweils einschließlich) liegt.*

Editieren der Kommandozeile

Elementare Befehle im Emacs-Modus

Befehl	Beschreibung
CTRL-B	Cursor ein Zeichen zurück bewegen (ohne zu löschen).
CTRL-F	Cursor ein Zeichen vorwärts bewegen.
DEL	Zeichen links vom Cursor löschen.
CTRL-D	Zeichen unter dem Cursor löschen.

Wort-Befehle im Emacs-Modus

Befehl	Beschreibung
ESC b	Ein Wort zurück bewegen.
ESC f	Ein Wort nach vorn bewegen.
ESC DEL	Wort links vom Cursor löschen.
ESC d	Wort unter dem Cursor löschen.
CTRL-Y	Letzten gelöschten Text zurückholen („yank").

Zeilenbefehle im Emacs-Modus

Befehl	Beschreibung
CTRL-A	Cursor an den Zeilenanfang positionieren.
CTRL-E	Cursor an das Zeilenende positionieren.
CTRL-K	Bis zum Ende der Zeile löschen.

Befehle im Emacs-Modus, um durch die History zu navigieren

Befehl	Beschreibung
CTRL-P	Zur vorherigen Zeile bewegen.
CTRL-N	Zur nächsten Zeile bewegen.
CTRL-R	Nach hinten suchen.
ESC <	Zur ersten Zeile der History springen.
ESC >	Zur letzten Zeile der History springen.

Vervollständigungsbefehle

Befehl	Beschreibung
TAB	Versuche, den eingegebenen Text zu vervollständigen.
ESC ?	Mögliche Vervollständigungen aufführen.
ESC /	Versuche, den Dateinamen zu vervollständigen.
CTRL-X /	Führe die möglichen Vervollständigungen zu Dateinamen an.
ESC ~	Versuche, den Benutzernamen zu vervollständigen.
CTRL-X ~	Führe die möglichen Vervollständigungen zu Benutzernamen an.
ESC $	Versuche, die Variable zu vervollständigen.
CTRL-X $	Führe die möglichen Vervollständigungen zu Variablen an.
ESC @	Versuche, den Rechnernamen zu vervollständigen.
CTRL-X @	Führe die möglichen Vervollständigungen zu Rechnernamen an.
ESC !	Versuche, den Befehl zu vervollständigen.
CTRL-X !	Führe die möglichen Befehls-Vervollständigungen an.
ESC TAB	Versuche eine Vervollständigung anhand früher eingegebener Befehle.

Diverse weitere Befehle im Emacs-Modus

Befehl	Beschreibung
CTRL-J	Wie RETURN.
CTRL-L	Bildschirm löschen; aktuelle Zeile in der Nähe der Bildschirmoberkante positionieren.
CTRL-M	Wie RETURN.
CTRL-O	Wie RETURN, zeige danach die nächste Zeile der Befehls-History an.
CTRL-T	Vertausche die beiden Zeichen links vom und unter dem Cursor und bewege den Cursor ein Zeichen weiter nach vorn.
CTRL-U	Lösche die Zeile vom Zeilenanfang bis zum Zeichen links vom Cursor.
CTRL-V	Füge das nächste Zeichen direkt ein, anstatt es als Befehl zu interpretieren.
CTRL-[	Wie ESC (auf den meisten Tastaturen).
ESC c	Mache das erste Zeichen des Wortes nach dem Cursor groß. (Wenn der Cursor mitten im Wort steht, wird das Zeichen unter dem Cursor groß gemacht.)
ESC u	Ändere alle Zeichen im nachfolgenden Wort in die entsprechenden Großbuchstaben.
ESC l	Ändere alle Zeichen im nachfolgenden Wort in die entsprechenden Kleinbuchstaben.
ESC .	Füge das letzte Wort aus der vorherigen Kommandozeile rechts vom Cursor ein.
ESC _	Wie ESC.

Befehle im vi-Modus

Der vi-Modus kennt zwei Sub-Modi, den Einfügemodus und den Befehlsmodus. Der Standard-Modus ist der Einfügemodus; mit ESC kann zwischen den beiden Modi umgeschaltet werden. Außerdem gelangen Sie im Befehlsmodus durch Eingabe von **a** (append - anhängen) oder **i** (insert - einfügen) in den Einfügemodus.

Editierbefehle im vi-Einfügemodus

Befehl	Beschreibung
DEL	Vorheriges Zeichen löschen.
CTRL-W	Vorheriges Wort löschen.
CTRL-V	Eingegebenes Zeichen direkt einfügen, anstatt es als Befehl zu interpretieren.
ESC	Sprung zum Befehlsmodus (siehe nächster Abschnitt).

Grundlegende Befehle im vi-Befehlsmodus

Befehl	Beschreibung
h	Cursor ein Zeichen nach links bewegen.
l	Cursor ein Zeichen nach rechts bewegen.
b	Cursor ein Wort nach links bewegen.
w	Cursor ein Wort nach rechts bewegen.
B	Cursor an den Anfang des letzten nicht-leeren Wortes bewegen.
W	Cursor an den Anfang des nächsten nicht-leeren Wortes bewegen.
e	Cursor zum Ende des aktuellen Wortes bewegen.
E	Cursor zum Ende des aktuellen nicht-leeren Wortes bewegen.
0	Cursor an den Zeilenanfang setzen.
^	Cursor auf das erste nicht-leere Zeichen setzen.
$	Cursor an das Zeilenende setzen.

Befehle, um in den vi-Eingabemodus zu gelangen

Befehl	Beschreibung
i	Text vor dem aktuellen Zeichen einfügen (insert).
a	Text hinter dem aktuellen Zeichen einfügen (append).
I	Text am Zeilenanfang einfügen.
A	Text am Zeilenende einfügen.
R	Existierenden Text überschreiben.

Einige Befehle zum Löschen von Text im vi-Modus

Befehl	Beschreibung
dh	Zeichen links vom Cursor löschen.
dl	Zeichen unter dem Cursor löschen.
db	Bis zum nächsten Wortanfang nach links löschen.
dw	Bis zum nächsten Wortende nach rechts löschen.
dB	Bis zum nächsten Anfang eines nicht-leeren Wortes nach links löschen.
dW	Bis zum nächsten Ende eines nicht-leeren Wortes nach rechts löschen.
d$	Bis zum Zeilenende nach rechts löschen.
d0	Bis zum Zeilenanfang nach links löschen.

Abkürzungen für Lösch-Befehle im vi-Modus

Befehl	Beschreibung
D	Wie **d$** (bis zum Zeilenende löschen).
dd	Wie **0d$** (ganze Zeile löschen).
C	Wie **c$** (bis zum Zeilenende löschen und in den Eingabemodus wechseln).
cc	Wie **0c$** (ganze Zeile löschen und in den Eingabemodus wechseln).
X	Wie **dl** (Zeichen links vom Cursor löschen).
x	Wie **dh** (Zeichen unter dem Cursor löschen).

Befehle im vi-Befehlsmodus zum Durchsuchen der Befehls-History

Befehl	Beschreibung
k oder -	Eine Zeile zurück.
j oder +	Eine Zeile nach vorn.
G	Zur Zeile springen, die durch das numerische Argument angegeben ist.
/*string*	Vorwärts nach *string* suchen.
?*string*	Rückwärts nach *string* suchen.
n	In der gleichen Richtung weitersuchen.
N	In der entgegengesetzten Richtung weitersuchen.

Befehle im vi-Modus, um einzelne Zeichen zu finden

Befehl	Beschreibung
f*x*	Cursor rechts vom nächsten *x* positionieren.
F*x*	Cursor links vom letzten *x* positionieren.
t*x*	Cursor rechts vom nächsten *x* positionieren, danach ein Zeichen nach links gehen.
T*x*	Cursor links vom nächsten *x* positionieren, danach ein Zeichen nach rechts gehen.
;	Letzten Befehl aus diesem Block wiederholen.
,	Letzten Befehl aus diesem Block in entgegengesetzter Richtung wiederholen.

Weitere Befehle im vi-Modus

Befehl	Beschreibung
~	Groß-/Kleinschreibung des aktuellen Zeichens umschalten.
_	Das letzte Wort des vorherigen Befehls anhängen und danach in den Eingabemodus schalten.
CTRL-L	Bildschirm löschen und neu zeichnen; praktisch, wenn der Bildschirminhalt durcheinander geraten ist.
#	Das #-Zeichen (leitet einen Kommentar ein) der Zeile voranstellen und die gesamte Zeile in die History-Datei schreiben; dies ist nützlich, um einen Befehl später noch einmal ausführen zu können, ohne ihn neu tippen zu müssen.

Quoten

Quoten bedeutet, die besondere Bedeutung eines Zeichens vorübergehend abzuschalten und statt dessen das Zeichen selbst zu verwenden. Die folgenden Zeichen werden in der **bash** gesondert behandelt:

;	Befehlstrenner
&	Hintergrundausführung
( )	Gruppieren von Befehlen (erzeugt eine Sub-Shell)
{ }	Gruppieren von Befehlen zu einem Block
|	Pipe
> < &	Symbole zur Umleitung
* ? [ ] ~ !	Platzhalterzeichen in Dateinamen
" ' \	Zeichen zum Quoten anderer Zeichen
`	Befehlssubstitution
$	Variablen- oder Befehlssubstitution
Zeilenumbruch **Leerzeichen** **Tabulator**	Trennzeichen zwischen zwei Wörtern
#	Kommentar

Die folgenden Zeichen können zum Quoten verwendet werden:

" " Alles zwischen " und " wird wortwörtlich interpretiert, ausgenommen die folgenden Zeichen, die ihre besondere Bedeutung behalten:

- $ Variablen-Substitution
- ` Befehlssubstitution
- " Das Ende des gequoteten Abschnitts

' ' Alles zwischen ' und ' wird wortwörtlich interpretiert; ausgenommen davon ist nur ein weiteres ', das die Quotierung wieder aufhebt.

\ Das auf den \ folgende Zeichen wird wortwörtlich interpretiert. Dies kann innerhalb von " " verwendet werden, um ", $ und ` zu quoten, und wird häufig zum Quoten von weiteren \, Leerzeichen oder Zeilenumbrüchen verwendet.

Beispiele

```
$ echo 'Einfache Anführungsstriche "schützen" doppelte Anführungsstriche
Einfache Anführungsstriche "schützen" doppelte Anführungsstriche

$ echo "Ist das nicht etwas \"Besonderes\"?"
Ist das nicht etwas "Besonderes"?

$ echo "Es gibt `ls|wc -l` Dateien in `pwd`"
Es gibt 43 Dateien in /home/kalle

$ echo "Der Wert von \$x ist $x"
Der Wert von $x ist 100
```

Schreibweisen von Befehlen

`Befehl &`	Führt `Befehl` im Hintergrund aus.
`Befehl1 ; Befehl2`	Befehlsfolge; führt mehrere `Befehle` in derselben Zeile aus.
`(Befehl1 ; Befehl2)`	Starten einer Sub-Shell; `Befehl1` und `Befehl2` werden als Befehlsgruppe betrachtet.
`Befehl1 | Befehl2`	Pipe; die Ausgabe von `Befehl1` wird als Eingabe von `Befehl2` verwendet.
``Befehl1 `Befehl2` ``	Befehlssubstitution; die Ausgabe von `Befehl2` wird als Argument(e) für `Befehl1` verwendet.
`Befehl1 $(Befehl2)`	Befehlssubstitution; Schachtelung ist zulässig.
`Befehl1 && Befehl2`	AND; `Befehl2` wird nur ausgeführt, wenn `Befehl1` erfolgreich war.
`Befehl1 || Befehl2`	OR; `Befehl2` wird nur ausgeführt, wenn `Befehl1` fehlgeschlagen ist.
`{ Befehl1 ; Befehl2 }`	Die Befehle werden in der laufenden Shell ausgeführt.

bash

Beispiele

```
$ nroff Datei &                              Formatieren im Hintergrund.
$ cd; ls                                     Nacheinander ausführen.
$ (date; who; pwd) > logfile                 Die gesamte Ausgabe wird umgeleitet.
$ sort file | pr -3 | lp                     Eine Datei sortieren, für den Druck
                                             aufbereiten und drucken.
$ vi `grep -l ifdef *.c`                     Alle von grep gefundenen Dateien editieren.
$ egrep '(yes|no)' `cat list`                Angabe einer Liste von Dateien, die
                                             durchsucht werden soll.
$ egrep '(yes|no)' $(cat list)               bash-Schreibweise des gleichen Befehls.
$ egrep '(yes|no)' $(<list)                  Das gleiche, aber schneller.
$ grep XX file && lp file                    Datei drucken, wenn sie ein bestimmtes
                                             Muster enthält.
$ grep XX file || echo "XX not found"        Fehlermeldung ausgeben, wenn das
                                             Muster nicht gefunden wurde.
```

Schreibweisen der Umleitung

Datei-Deskriptor	Name	Übliche Abkürzung	Üblicher Default
0	Standardeingabe	stdin	Tastatur
1	Standardausgabe	stdout	Bildschirm
2	Standardfehler	stderr	Bildschirm

Die normalen Ein- und Ausgabewege können folgendermaßen geändert werden:

I/O-Umleitungen

Umleitezeichen	Bedeutung
> *Datei*	Standardausgabe nach *Datei* umleiten.
< *Datei*	Standardeingabe nach *Datei* umleiten.
Befehl1 | *Befehl2*	Pipe; Standardausgabe von *Befehl1* als Standardeingabe von *Befehl2* verwenden.
>> *Datei*	Standardausgabe nach *Datei* umleiten; an *Datei* anhängen, wenn die Datei schon existiert.
>| *Datei*	Umleitung der Standardausgabe nach *Datei* erzwingen, auch wenn **noclobber** gesetzt ist.
n>|	Ausgabe vom Datei-Deskriptor auf *Datei* erzwingen, selbst wenn **noclobber** gesetzt ist.
<> *Datei*	*Datei* sowohl als Standardeingabe als auch als Standardausgabe verwenden.
<< *text*	Von der Standardeingabe lesen, bis *text* gefunden wird. (*text* kann in einer Shell-Variablen abgespeichert sein.) Die Eingabe wird typischerweise an der Konsole eingegeben oder steht in einem Shell-Skript. Diese Schreibweise wird oft bei den Befehlen **cat**, **echo**, **ex**, und **sed** benutzt. Wenn *text* in Quotes eingeschlossen ist, wird auf den Daten der Standardeingabe keine Variablen- oder Befehlssubstitution etc. vorgenommen.
n> *Datei*	Datei-Deskriptor *n* auf *Datei* umleiten.
n< *Datei*	*Datei* zum Datei-Deskriptor *n* machen.
>&*n*	Standardausgabe duplizieren und auch auf den Datei-Deskriptor *n* ausgeben.
<&*n*	Standardeingabe duplizieren und auch vom Datei-Deskriptor *n* lesen.
&>*Datei*	Standardausgabe und Standardfehler nach *Datei* umlenken.
<&-	Standardeingabe schließen.
>&-	Standardausgabe schließen.
n>&-	Ausgabekanal des Datei-Deskriptors *n* schließen.
n<&-	Eingabekanal des Datei-Deskriptors *n* schließen.

Beispiele

```
$ cat teil1 > buch
$ cat teil2 teil3 >> buch
$ mail elke < bericht

$ sed 's/^/XX /' << END_ARCHIVE
> So werden haeufig Shell-Archive "eingepackt",
> um weitergegeben werden zu können. Normalerweise wuerde
> sed aus einem Shell-Skript heraus gestartet werden
> und nicht von der Kommandozeile.
> END_ARCHIVE
XX So werden haeufig Shell-Archive "eingepackt",
XX um weitergegeben werden zu können. Normalerweise wuerde
XX sed aus einem Shell-Skript heraus gestartet werden
XX und nicht von der Kommandozeile.
```

Umlenken der Standardausgabe nach Standardfehler:

```
$ echo "Falscher Aufruf: Fragen Sie ihren Systemverwalter" 1>&2
```

Der folgende Befehl schickt die Ausgabe (Dateien gefunden) nach *dateiliste* und die Fehlermeldungen (nicht zugreifbare Dateien) in die Datei *kein_zugriff*:

```
$ find / -print > dateiliste 2>kein_zugriff
```

Koprozesse

Koprozesse gibt es nur in der **bash**.

Befehl1 | *Befehl2* |&	Koprozeß; führt die Pipeline im Hintergrund aus. Die Shell errichtet eine bidirektionale Pipe, womit sowohl Standardeingabe als auch Standardausgabe umgeleitet werden können.
read -p *var*	Übernimmt die Eingabe des Koprozesses in die Variable *var*.
print -p *string*	Schickt die Zeichenkette *string* an den Koprozeß.
Befehl **<&p**	Holt die Eingabe für den Befehl *Befehl* vom Koprozeß.
Befehl **>&p**	Schickt die Eingabe des Befehls *Befehl* zum Koprozeß.

Beispiele

```
cat memo                                 Dateiinhalt ausgeben
Wer begonnen hat, der hat
schon halb vollendet. (Horaz)
ed - memo |&                             Koprozeß
print -p /halb/                          ed-Befehl an den Koprozeß schicken
read -p search                           Ausgabe des ed-Befehls in die Variable search übernehmen
print "$search"                          Zeile auf der Standardausgabe ausgeben
schon halb vollendet. (Horaz)
```

Variablen

Dieser Abschnitt beschreibt:

- Variablen-Substitution
- Eingebaute Shell-Variablen

Variablen-Substitution

Substitutions-Operatoren

Operator	Substitution
${*Varname*:-*Wort*}	Wenn *Varname* existiert und nicht leer ist, wird der Wert von *Varname* zurückgegeben, ansonsten *Wort*.
Einsatz:	Rückgabe eines Default-Wertes, wenn die Variable undefiniert ist.
Beispiel:	**${zaehler:-0}** ergibt 0, wenn **zaehler** undefiniert ist.

Operator	Substitution
${*Varname*:=*Wort*}	Wenn *Varname* existiert und nicht leer ist, wird der Wert von *Varname* zurückgegeben; ansonsten wird der Wert der Variablen auf *Wort* gesetzt und dieser Wert zurückgegeben. Positions- und andere besondere Parameter können nicht auf diese Art und Weise zugewiesen werden.
Einsatz:	Eine Variable auf einen Default-Wert setzen, wenn sie undefiniert ist.
Beispiel:	**${zaehler:=0}** setzt **zaehler** auf 0, wenn die Variable undefiniert ist.
${*Varname*:?*Nachricht*}	Wenn *Varname* existiert und nicht leer ist, wird der Wert von *Varname* zurückgegeben. Ansonsten wird *Varname*:, gefolgt von *Nachricht* ausgegeben und der aktuelle Befehl oder das aktuelle Skript beendet (nur in nicht-interaktiven Shells). Wenn *Nachricht* weggelassen wird, wird die Default-Meldung **parameter null or not set** ausgegeben.
Einsatz:	Abfangen von Fehlern, die von undefinierten Variablen herrühren.
Beispiel:	**{zaehler:?"undefiniert!"}** gibt "zaehler: undefiniert!" aus und bricht ab, wenn **zaehler** undefiniert ist.
${*Varname*:+*Wort*}	Wenn *Varname* existiert und nicht leer ist, wird *Wort* zurückgegeben, ansonsten 0.
Einsatz:	Die Existenz einer Variablen überprüfen.
Beispiel:	**${zaehler:+1}** gibt 1 zurück (was für "wahr" stehen könnte), wenn **zaehler** definiert ist.

Operatoren zum Mustervergleich

Operator	Bedeutung
${*Variable#Muster*}	Wenn das Muster auf den Anfang des Wertes der Variablen paßt, wird der kürzeste passende Teil gelöscht und der Rest zurückgegeben.
${*Variable##Muster*}	Wenn das Muster auf den Anfang des Wertes der Variablen paßt, wird der längste passende Teil gelöscht und der Rest zurückgegeben.
${*Variable%Muster*}	Wenn das Muster auf das Ende des Wertes der Variablen paßt, wird der kürzeste passende Teil gelöscht und der Rest zurückgegeben.
${*Variable%%Muster*}	Wenn das Muster auf das Ende des Wertes der Variablen paßt, wird der längste passende Teil gelöscht und der Rest zurückgegeben.

Vordefinierte Shell-Variablen

Vordefinierte Variablen werden automatisch von der Shell gesetzt und normalerweise in Shell-Skripten verwendet. Vordefinierte Variablen können die oben beschriebenen Substitutions-Verfahren werden. Die Variablen werden hier ohne die Operatoren gezeigt, diese werden aber gleichwohl immer benötigt, wenn die Variablen dereferenziert werden sollen.

Grundlegende Shell-Optionen

Option	Beschreibung
emacs	Wechsel zum Emacs-Editiermodus (Default).
ignoreeof	Wenn diese Variable gesetzt ist, reicht ein einzelnes **CTRL-D** nicht zum Abmelden; es muß ausdrücklich der **exit**-Befehl verwendet werden. Der gleiche Effekt kann durch Setzen der Shell-Variablen IGNOREEOF auf 1 erzielt werden.
noclobber	Verhindert das Überschreiben bereits vorhandener Dateien durch Ausgabeumleitung (>).
noglob	Schaltet das Expandieren von Dateinamen mit Platzhalterzeichen wie * und ? ab. (Expansion von Platzhalterzeichen wird manchmal auch *globbing* genannt.)
nounset	Wenn eine nicht definierte Variable verwendet werden soll, wird eine Fehlermeldung ausgegeben.
vi	Wechsel zum **vi**-Editiermodus.

bash

Standard-Variablen

Variable	Bedeutung
COLUMNS	Anzahl der Spalten des Bildschirms oder Bildschirmfensters.
EDITOR	Pfadname des Editors.
LINES	Anzahl der Zeilen des Bildschirms oder Bildschirmfensters.
SHELL	Pfadname der gerade laufenden Shell.
TERM	Verwendeter Terminal-Typ.

History-Variablen

Variable	Bedeutung
HISTCMD	Die Nummer des aktuellen Befehls in der History.
HISTCONTROL	Wenn HISTCONTROL auf den Wert **ignorespace** gesetzt ist, werden Zeilen, die mit einem Leerzeichen beginnen, nicht in die History aufgenommen. Wenn der Wert **ignoredups** ist, werden Zeilen, die mit der letzten Zeile der History identisch sind, nicht in die History aufgenommen. Mit **ignoreboth** können beide Optionen verwendet werden.
HISTFILE	Größe der History-Datei, die die Editier-Modi verwenden.
HISTFILESIZE	Maximale Anzahl von Zeilen in der History-Datei. Der Defaultwert ist 500.
HISTSIZE	Maximale Anzahl von Befehlen, die in der Befehls-History gespeichert werden. Der Defaultwert ist 500.
FCEDIT	Pfadname des Editors, der zusammen mit dem **fc**-Befehl verwendet werden soll.

Mail-Variablen

Variable	Bedeutung
MAIL	Name der Datei, die auf eingehende Nachrichten überprüft werden soll.
MAILCHECK	Wie oft (in Sekunden) nach neuen Nachrichten geschaut wird (der Defaultwert ist 60 Sekunden).
MAILPATH	Durch Doppelpunkte (`:`) getrennte Liste von Dateinamen, die auf eingehende Nachrichten überprüft werden sollen.

Status-Variablen

Variable	Bedeutung
HOME	Name des Home-Verzeichnisses.
SECONDS	Sekunden, die seit dem Aufruf der aktuellen Shell vergangen sind.
BASH	Pfadname der aktuellen Shell.
BASH_VERSION	Die Versionsnummer der aktuellen Shell.
PWD	Aktuelles Verzeichnis.
OLDPWD	Verzeichnis vor dem letzten **cd**-Befehl.

Arithmetische Ausdrücke

Mit dem **let**-Befehl können Ganzzahl-Rechnungen durchgeführt werden. Die **bash** bietet Möglichkeiten, Ganzzahl-Werte zu substituieren (sowohl als Argumente von Befehlen als auch als Werte für Variablen); außerdem können Zahlen in verschiedene Zahlensysteme konvertiert werden:

`$(( ` *`Ausdr`* ` ))`	Benutze den Wert des arithmetischen Ausdrucks in Klammern.

Operatoren

Die **bash** verwendet die gleichen arithmetischen Operatoren wie die Programmiersprache C, die in der folgenden Liste in absteigender Präzedenz geordnet sind. Durch Setzen von Klammern können die Präzedenzen entsprechend geändert werden.

-	Unäres Minus
! ~	Logische Negation; binäres Inverse (Einerkomplement)
* / %	Multiplikation; Division; Modulus (Rest)
+ -	Addition; Subtraktion
<< >>	Bitweises Schieben nach links; bitweises Schieben nach rechts
<= >=	Kleiner oder gleich; größer oder gleich
< >	Kleiner als; größer als
== !=	Gleichheit; Ungleichheit (jeweils von links nach rechts ausgewertet)
&	Bitweises UND
^	Bitweises Exklusiv-ODER
|	Bitweises ODER

&&	Logisches UND
||	Logisches ODER
=	Wertzuweisung
+= -=	Addition bzw. Subtraktion mit anschließender Zuweisung
*= /= %=	Multiplikation, Division bzw. Restbildung mit anschließender Zuweisung
&= ^= |=	Bitweises AND, XOR bzw. OR mit anschließender Zuweisung
<<= >>=	Bitweises Schieben nach links bzw. rechts mit anschließender Zuweisung

Beispiele

Beispiele und weitere Informationen zu arithmetischen Ausdrücken finden Sie bei der Beschreibung des **let**-Befehls.

```
let "zaehler=0" "i = i + 1"    Zuweisung von i und zaehler
let "num % 2"                  Test, ob eine Zahl gerade ist
```


Befehls-History

Es ist in der **bash** möglich, früher eingegebene Befehle anzeigen zu lassen oder sie zu verändern, ähnlich wie der History-Mechanismus in der C-Shell. Befehle, die in der History-Liste stehen, können folgendermaßen verändert werden:

- Mit dem Editiermodus
- Mit dem **fc**-Befehl

Außerdem funktionieren die Befehls-Substitutionen, die in Kapitel 4, *bash: Die Bourne Again Shell*, beschrieben werden, auch in der **bash**.

Editiermodus

Der Editiermodus emuliert viele Funktionen der Editoren **vi** und Emacs. Die History-Liste wird wie eine Datei behandelt. Nach Aufruf des Editors können Sie mit Tastaturbefehlen zu der Kommandozeile gelangen, die Sie ausführen wollen. Außerdem können Sie diese Zeile vor der Ausführung auch noch verändern. Wenn Sie den Befehl ausführen möchten, drücken Sie auf RETURN.

Der Editiermodus kann auf verschiedene Arten und Weisen aufgerufen werden. Die folgenden Beispiele sind äquivalent:

```
$ VISUAL=vi
$ EDITOR=vi
$ set -o vi               Hat Vorrang vor VISUAL und EDITOR
```

Beachten Sie, daß die **vi**-Emulation im Eingabemodus startet; drücken Sie auf ESCAPE, um in den **vi**-Befehlsmodus zu gelangen.

vi	Emacs	Ergebnis
`k`	`CTRL-P`	Vorherigen Befehl holen.
`j`	`CTRL-N`	Nächsten Befehl holen.
`/`*`String`*	`CTRL-R` *`String`*	Nächsten Befehl holen, der *String* enthält.
`h`	`CTRL-B`	Cursor ein Zeichen zurückbewegen.

vi	Emacs	Ergebnis
`l`	`CTRL-F`	Cursor ein Zeichen nach vorn bewegen.
`b`	`ESC-B`	Cursor ein Wort zurückbewegen.
`w`	`ESC-F`	Cursor ein Wort nach vorn bewegen.
`X`	`DEL`	Letztes Zeichen löschen.
`x`	`CTRL-D`	Ein Zeichen löschen.
`dw`	`ESC-D`	Vorheriges Wort löschen.
`db`	`ESC-H`	Letztes Wort löschen.
`xp`	`CTRL-T`	Zwei Zeichen vertauschen.

Der fc-Befehl

Mit fc -l können Sie Befehle aus der History ausgeben und mit -e editieren. Der Eintrag zu fc bei der Beschreibung der eingebauten Befehle enthält nähere Informationen.

Beispiele

`$ history`	*Letzte 16 Befehle ausgeben.*
`$ fc -l 20 30`	*Befehle 20 bis 30 ausgeben.*
`$ fc -l -5`	*Letzte 5 Befehle ausgeben.*
`$ fc -l cat`	*Letzte Kommandozeile ausgeben, die mit* cat *anfängt.*
`$ fc -ln 5 > zeile`	*Befehl 5 in Datei* zeile *speichern.*
`$ fc -e vi 5 20`	*Befehle 5 bis 20 mit dem* vi *editieren.*
`$ fc -e emacs`	*Letzten Befehl mit dem Emacs bearbeiten.*
`$ !!`	*Letzten Befehl noch einmal ausführen.*
`$ !cat`	*Letzten* cat*-Befehl noch einmal ausführen.*
`$ !cat datei`	*Letzten Befehl noch einmal ausführen, wobei* datei *an das Ende der Argumentliste angehängt wird.*

Befehls-Substitution

!	History-Substitution anfangen.
!!	Letzter Befehl.
!*N*	Befehl Nummer *N* in der History-Liste.
!-*N*	*N*t-letzter Befehl.
!*String*	Jüngster Befehl, der mit *String* anfängt.
!?*String*?	Jüngster Befehl, der *String* enthält.
!?*String*?%	Jüngstes Befehl-Argument, das *String* enthält.
!$	Letztes Argument des letzten Befehls.
!!*String*	Letzter Befehl, an den *String* angehängt wird.
!*N String*	Befehl Nummer *N*, an den *String* angehängt wird.
!{*s1*}*s2*	Letzter Befehl, der mit *s1* anfing und an den *s2* angehängt wird.
^*alt*^*neu*^	Schnell-Ersetzung: Ersetzung von *alt* durch *neu* im letzten Befehl und Ausführung des neuen Befehls.

Eingebaute Befehle

Beispiele, die auf der Kommandozeile eingegeben werden sollen, erscheinen hier mit dem $-Prompt. Alle anderen Beispiele sind Programm-Fragmente, die in Shell-Skripten verwendet werden können. Zur besseren Übersicht werden hier auch einige der reservierten Schlüsselwörter aufgeführt, die in Befehlen verwendet werden, die sich über mehrere Zeilen erstrecken.

— *#*

Alles, was auf derselben Zeile folgt, wird ignoriert. # wird in Shell-Skripten als kommentareinleitendes Zeichen verwendet und ist kein Befehl im eigentlichen Sinne.

#!Shell — #!

Wird als erste Zeile in einem Shell-Skript verwendet und bedeutet, daß das Shell-Skript von der Shell *Shell* (mit optionalen Argumenten) ausgeführt werden soll. Beispiel:

```
#!/bin/bash
```

: — :

Leerer Befehl. Gibt einen Ausführungsstatus von 0 zurück. Wird manchmal als erstes Zeichen in einer Datei verwendet, um anzuzeigen, daß es sich um ein **bash**-Skript handelt. Shell-Variablen können nach dem : aufgeführt werden und werden dann zu ihren jeweiligen Werten expandiert.

Beispiel

Test, ob jemand angemeldet ist:

```
if who | grep $1 > /dev/null
then :                  # tue nichts,
                        # wenn Muster gefunden wurde
else echo "Benutzer $1 ist nicht angemeldet."
fi
```

. *Datei* [*Argumente*] — .

Siehe **source**.

---

alias [*Name*[='*Befehl*']] — **alias**

Erzeugt einen Kurznamen (Aliasnamen) *Name* als Synonym für *Befehl.* Wenn ='*Befehl*' weggelassen wird, wird ausgegeben, wofür *Name* steht, wenn auch *Name* weggelassen wird, werden alle Aliasnamen ausgegeben. Wenn hinter *Befehl* ein Leerzeichen steht, werden die folgenden Kommandozeilen-Argumente auf weitere Aliasnamen überprüft. Siehe auch **unalias**.

bg	**bg** [*JobIDs*] Setzt den aktuellen Job oder *JobIDs* in den Hintergrund. Siehe „Job-Verwaltung" am Ende dieses Kapitels.
bind	**bind** [*Optionen*] **bind** [*Optionen*] *Tasten*:*Funktion* Ausgeben oder Setzen einer Tasten-/Funktionsbindung. *Optionen* -m *keymap* Gibt die Tastaturtabelle an, die in weiteren Bindungen verwendet werden soll. Mögliche Tastaturtabellen sind: **emacs**, **emacs-standard**, **emacs-meta**, **emacs-ctlx**, **vi**, **vi-move**, **vi-command**, und **vi-insert**. -l Ausgeben aller Funktionen zum Editieren der Kommandozeile. -v Ausgeben aller Funktionsnamen mit ihren Bindungen (wenn vorhanden). -d Ausgabe der Funktionsnamen mit ihren Bindungen in einem Format, das wieder eingelesen werden kann. -f *Dateiname* Setzen der Bindungen in *Dateiname*. -q *Funktion* Ausgeben der Bindungen von *Funktion*.
break	**break** [*n*] Beendet die innerste (am tiefsten verschachtelte) oder die *n* innersten **for**-, **while**-, oder **until** Schleifen. Beendet außerdem die Befehlsabarbeitung in einem **select**-Befehl.
built-in	**built-in** *Befehl* [*Argumente*] Ausführen des eingebauten *Befehls*. Dies ist nützlich, wenn eingebaute Befehle innerhalb von Skripten desselben Namens aufgerufen werden sollen.

case	**case** *String in*
	  RegAusdr)
	  Befehle
	   ;;
	   ...
	esac
	
	Wenn *String* auf den regulären Ausdruck *RegAusdr* paßt, werden die *Befehle* ausgeführt. Es werden alle regulären Ausdrücke überprüft, bis ein passender gefunden wird. Um alle Zeichenketten, die auf keinen regulären Ausdruck passen, abfangen zu können, kann * als letzter regulärer Ausdruck verwendet werden.

cd [*Verzeichnis*] cd

Wenn kein Argument angegeben ist, wird zum Home-Verzeichnis des Benutzers gewechselt, ansonsten zum *Verzeichnis*. Wenn *Verzeichnis* ein relativer Pfadname ist, aber im aktuellen Verzeichnis nicht gefunden wird, wird die CDPATH-Variable durchsucht.

command [*Optionen*] *Befehl* [*Argumente*] command

Ausführung des Befehls *Befehl*, wobei nicht nach Funktionen gesucht wird, d.h., es wird kein Befehl ausgeführt, der weder eingebaut noch im durch PATH bezeichneten Pfad ist. Der Ausführungsstatus wird auf den von *Befehl* zurückgegebenen gesetzt. Wenn *Befehl* nicht gefunden werden konnte, ist der Ausführungsstatus 127.

- -p Die PATH-Variable wird ignoriert und nur der Default-Pfad durchsucht.
- -- Alles, was folgt, wird als Argument und nicht als Option interpretiert.

bash

continue [*n*] continue

Überspringen der verbleibenden Befehle in einer **for**-, **while**-, oder **until**-Schleife; die Abarbeitung wird mit dem nächsten Durchlauf der Schleife fortgesetzt. Wird der Parameter *n* angegeben, werden entsprechend viele Schleifen übersprungen.

declare [*Optionen*] [*Name*[=*Wert*]] declare
typeset [*Optionen*] [*Name*[=*Wert*]]

Ausgeben oder Setzen des Wertes von Variablen. Ein + anstelle des – dreht die Bedeutung ins Gegenteil um.

- -f Nur Funktionsnamen verwenden.
- -r Variablen dürfen später nicht verändert werden.
- -x Variablen werden exportiert.
- -i Interpretieren der Variablen als Ganzzahl und Evaluation ihrer Werte.

dirs [*Optionen*] dirs

Ausgeben der Verzeichnisse, die zur Zeit aufgrund von **pushd**/**popd**-Befehlen gemerkt werden.

Optionen

+*Eintrag*
  Eintrag Nr. *Eintrag* ausgeben (der erste Eintrag hat die Nummer 0).

-*Eintrag*
  Eintrag Nr. *Eintrag* vom Ende her ausgeben.

-l Lange Ausgabe.

echo

echo [*Optionen*] [*String*]

Ausgeben von *String* und eines Zeilenwechsels. Wenn *String* nicht angegeben wird, wird nur der Zeilenwechsel ausgegeben. In der **bash** ist **echo** lediglich ein Aliasname für **print -**. (Siehe auch **echo** in Kapitel 1, *Linux-Benutzerbefehle*.)

- **-n** Keinen Zeilensprung ausgeben.
- **-e** Escape-Sequenzen interpretieren:

\a	Akustische Warnung.
\b	Löschen des letzten Zeichens.
\c	Unterdrücken des Zeilensprungs (wie **-n**).
\f	Seitenvorschub.

enable

enable [*–n*] [*–a*|*–all*] [*Befehl* ...]

Einschalten (oder Abschalten, wenn *-n* angegeben ist) von eingebauten Shell-Befehlen. Wenn *Befehl* nicht angegeben ist, werden die eingeschalteten eingebauten Befehle ausgegeben; mit *-a* oder *-all* werden alle eingebauten Befehle zusammen mit ihrem Status ausgegeben. Eingebaute Shell-Befehle können abgeschaltet werden, damit eigene Funktionen desselben Namens definiert werden können.

eval

eval [*Befehl Argumente*...]

Ausführen des Befehls *Befehl* mit den Argumenten *Argumente*.

exec

exec [[–] *Befehl*]

Ausführen des Befehls *Befehl* anstelle des aktuellen Prozesses (anstatt einen neues Prozeß zu erzeugen). **exec** ist auch nützlich für das Öffnen, Schließen und Kopieren von Datei-Deskriptoren. Wenn – angegeben wird, wird – als nulltes Argument an den neuen Prozeß übergeben.

Beispiele

`trap 'exec 2>&-' 0`	*Standardausgabe schließen, wenn die Ausführung des Shell-Skripts abgeschlossen ist (Signal 0).*
`$ exec /bin/csh`	*Aktuelle Shell durch die C-Shell ersetzen.*
`$ exec < infile`	*Standardeingabe auf infile umstellen.*

exit

exit [*n*]

Beenden eines Shell-Skripts mit dem Ausführungsstatus *n* (z.B. **exit 1**). *n* kann entweder 0 (erfolgreiche Ausführung) oder von 0 verschieden (Fehler) sein. Wenn *n* nicht angegeben wird, ist der Ausführungsstatus derjenige des letzten Befehls. **exit** kann auf der Kommandozeile verwendet werden, um ein Fenster zu schließen (Abmelden).

exit

Beispiel

```
if [ $# -eq 0 ]; then
   echo "Format:  $0 [-c] [-d] Datei(en)"
   exit 1              # Rückmeldung eines Fehlers
fi
```

export

export [*Optionen*] [*Variablen*]
export [*Optionen*] [*Name*=[*Wert*]] . . .

Weitergeben (Exportieren) einer oder mehrerer Shell-Variablen, wodurch diese Variablen eine globale Bedeutung erhalten (der Default ist lokale Bedeutung). Wenn beispielsweise der Wert einer Variablen, die in einem Shell-Skript definiert worden ist, von einem anderen von diesem Skript aufgerufenen Programm verwendet werden soll, muß diese Variable exportiert werden. Wenn keine Variablen angegeben werden, zeigt **export** alle Variablen an, die zur Zeit von der aktuellen Shell exportiert werden. Wenn *Name* und *Wert* angegeben sind, wird *Wert* der Variablen *Name* zugewiesen.

bash

Optionen

-- Alle folgenden Zeichenketten werden als Argumente und nicht als Optionen behandelt.

-f Alle Variablen werden als Funktionen erwartet.

-n Aufheben des Exportierens einer Variablen.

-p Ausgeben der Variablen, die zur Zeit von der aktuellen Shell ausgegeben werden.

fc

fc [*Optionen*] [*Erster*] [*Letzter*]
fc -e - [*alt*=*neu*] [*Befehl*]

Anzeigen oder Editieren von Befehlen in der History-Liste (**-l** und **-e** können nicht zusammen verwendet werden). **fc** stellt Möglichkeiten ähnlich denen des **history**-Befehls bzw. der !-Schreibweise in der C-Shell bereit. *Erster* und *Letzter* bezeichnen Zahlen oder Zeichenketten, mit denen die Befehle, die angezeigt oder editiert werden sollen, ausgewählt werden. Wenn *Letzter* weggelassen wird, wird **fc** auf einen einzigen Befehl angewendet, der durch *Erster* angegeben wird. Wenn sowohl *Erster* als auch *Letzter* weggelassen werden, editiert **fc** den letzten Befehl oder zeigt die letzten 16 an. Die zweite Form des **fc**-Befehls entnimmt der History einen Befehl, ersetzt die Zeichenkette *alt* durch die Zeichenkette *neu* und führt den so veränderten Befehl aus. Wenn keine Zeichenketten angegeben werden, wird der Befehl unverändert erneut ausgeführt. Wenn auch kein Befehl angegeben wird, wird der letzte Befehl erneut ausgeführt. *Befehl* kann wie *Erster* als Zahl oder als Zeichenkette angegeben werden. Beispiele finden Sie unter „Befehls-History".

Optionen

-e [*Editor*]
Aufrufen des Editors *Editor* um die angegebenen Befehle zu editieren. Mit der Shell-Variablen FCEDIT kann der Default-Editor eingestellt werden.

→

fc ←	-l [*Erster Letzter*] Ausgeben des oder der angegebenen Befehls/e oder der letzten 16, wenn keine Argumente angegeben sind. -n Unterdrücken der Befehlsnumerierung bei der Ausgabe mit -l. -r Umdrehen der Reihenfolge bei der Ausgabe mit -l. -s *Muster=NeuesMuster* Editieren der Befehle, indem jedes Auftreten von *Muster* durch *NeuesMuster* ersetzt wird. Danach werden die Befehle erneut ausgeführt.
fg	fg [*JobIDs*] Bringt den aktuellen Job oder *JobIDs* in den Vordergrund. Siehe „Job-Verwaltung“.

for	for *x* [in *Liste*]
	do
	    Befehle
	done
	
	Der Reihe nach wird jedes Wort in *Liste x* zugewiesen und ausgeführt. Wenn *Liste* weggelassen wird, werden die Positionsparameter $@ verwendet.
	
	Beispiele
	Durchnumerieren aller Dateien im aktuellen Verzeichnis und Abspeichern der erzeugten Dateien.
	
	`for $datei in * do`
	`     pr $datei > $datei.tmp`
	`done`
	
	Durchsuchen von Kapiteln nach einer Liste von Wörtern (wie **fgrep -f**):
	
	`for element in `cat programm_liste``
	`do`
	`     echo "Durchsuchen der Kapitel nach"`
	`     echo "Referenzen auf Programm $element..."`
	`     grep -c "$element.[co]" kap*`
	`done`

function	function *Befehl* { Definieren einer Funktion. Argumente werden verwendet wie Positionsparameter in einem Shell-Skript (**$1** usw.). Die Funktionsdefinition wird mit } abgeschlossen.
getopts	getopts *String Name* [*Argumente*] Verarbeiten der Kommandozeilen-Parameter (oder *Argumente*, falls angegeben) und Überprüfen auf zulässige Optionen. **getopts** wird in Schleifen in Shell-Skripten verwendet und soll eine konsistente Syntax von Kommandozeilen-Optionen

gewährleisten. *String* enthält die Optionsbuchstaben, die von **getopts** erkannt werden sollen. Gültige Optionen werden der Reihe nach verarbeitet und in der Shell-Variablen *Name* abgelegt. Wenn auf einen Optionsbuchstaben ein Doppelpunkt folgt, müssen der Option auf der Kommandozeile eines oder mehrere Argumente folgen. **getopts** verwendet die Shell-Variablen OPTARG und OPTIND.

getopts

---

hash [-r] [*Befehle*]

hash

Suchen nach *Befehle* und Merken des Verzeichnisses, in dem der Befehl gefunden wurde. Hashen erzeugt in der Shell eine Verbindung (Assoziation) zwischen einem Namen (z.B. eines Befehls) und einem absoluten Pfadnamen einer ausführbaren Datei, so daß bei zukünftigen Aufrufen nicht der PATH durchsucht werden muß. Wenn keine Argumente angegeben werden, gibt **hash** die aktuell gehashten Befehle aus. Dabei bedeutet *hits*, wie oft dieser Befehl von der aktuellen Shell aufgerufen wurde, und *cost*, wie teuer es war, den Befehl zu finden.

bash

---

help [*String*]

help

Ausgeben eines Hilfetexts zu allen eingebauten Befehlen oder zu denen, die auf *String* passen.

---

history [*Optionen*]
history [*Zeilen*]

history

Ausgabe einer numerierten Befehls-History, wobei veränderte Befehle mit einem * gekennzeichnet werden. Auch Befehle früherer Sitzungen werden mit aufgeführt. Mit dem Argument *Zeilen* kann angegeben werden, wie viele Zeilen ausgegeben werden sollen.

Optionen

- -a Die **bash** unterhält eine Datei *.bash_history* im Home-Verzeichnis des Benutzers, in der die Befehle früherer Sitzungen abgespeichert werden. Mit dieser Option werden die Befehle der aktuellen Sitzung an die Datei angehängt.
- -n Anhängen derjenigen Zeilen in der Datei *.bash_history* an die History-Liste, die dort noch nicht vorhanden sind.
- -r Verwenden der History-Liste in der Datei *.bash_history* anstelle der aktuellen.
- -w Überschreiben der Datei *.bash_history* mit der aktuellen History-List.

if

if *Test–Befehle*

Anfangen eines bedingten Ausdrucks. Mögliche Schreibweisen:

```
if Test-Befehle
then
    Befehle1
fi
```

```
if Test-Befehle
then
    Befehle1
else
    Befehle2
fi
```

```
if Test-Befehle
then
    Befehle1
elif Test-Befehle
then
    Befehle2
  ...
else
    Befehle
fi
```

Normalerweise führen die ersten **if**- und alle **elif**-Zeilen einen **test** oder **[]**-Befehl aus, obwohl auch alle anderen Befehle zulässig sind. Wenn **if** erfolgreich ist, d.h. das letzte der *Test-Befehle* einen Ausführungsstatus von 0 hat, wird *Befehle1* ausgeführt, ansonsten werden die folgenden **elif**-und **else**-Zeilen ausprobiert.

jobs

jobs [*Optionen*] [*JobIDs*]

Ausgabe aller laufenden und angehaltenen Jobs oder derjenigen, die durch *JobIDs* angegeben werden. So kann beispielsweise überprüft werden, ob ein langer Übersetzungsvorgang noch läuft. Auch vor dem Abmelden ist **jobs** nützlich. Siehe auch „Job-Verwaltung" am Ende dieses Kapitels.

Optionen

-l Ausgeben der Job-IDs und der Prozeßgruppen-IDs.

-n Ausgeben nur der Jobs, deren Status sich seit der letzten Benachrichtigung verändert hat.

-p Nur Prozeßgruppen-IDs ausgeben.

-x *Befehl* [*Argumente*]
Ausführen von *Befehl*. Wenn *JobIDs* angegeben werden, werden sie durch *Befehl* ersetzt.

kill

kill [*Optionen*] *IDs*

Beenden jedes durch Prozeß-*ID* oder Job-*ID* angegebenen Prozesses. Sie müssen der Besitzer des Prozesses oder Superuser sein.

Optionen

-*Signal*
Senden der Signalnummer *Signal*. Das Signal 9 kann vom Prozeß nicht abgefangen werden. Das Default-Signal ist TERM.

-- Alle folgenden Zeichenketten werden als Argumente und nicht als Optionen interpretiert.

-l Ausgabe der Signalnamen.

-s *Signal*
Angabe des gesendeten Signals. Hier kann sowohl der Signalname als auch die Signalnummer verwendet werden.

let *Ausdrücke*

let

Durchführung arithmetischer Ganzzahl-Berechungen, die durch *Ausdrücke* angegeben werden. *Ausdrücke* besteht aus Zahlen, Operatoren und Shell-Variablen (in denen der Dereferenzierungsoperator $ weggelassen werden kann). Die Ausdrücke müssen gequotet werden, wenn sie Leerzeichen oder andere Sonderzeichen enthalten. Weitere Informationen und Beispiele finden Sie im Abschnitt „Arithmetische Ausdrücke“ weiter vorn in diesem Kapitel und unter **expr** in Kapitel 1.

Beispiele

Die beiden folgenden Beispiele addieren 1 zum Wert der Variablen i und legen den neuen Wert wieder in i ab.

```
let i=i+1
let "i = i + 1"
```

local [*Variable*[=*Wert*]] [*Variable2*[=*Wert*]] ...

local

Wenn keine Argumente angegeben werden, werden alle lokalen Variablen ausgegeben. Anderenfalls wird die angegebene lokale Variable angelegt und ggf. ein Wert zugewiesen.

logout

logout

Verlassen der Shell. Kann nur verwendet werden, wenn es sich um eine Anmelde-Shell handelt; ansonsten muß **exit** verwendet werden.

popd [*Optionen*]

popd

Manipulieren des Verzeichnis-Stacks. Wenn keine Optionen angegeben werden, wird das oberste Verzeichnis vom Stack genommen und dorthin verzweigt.

Optionen

+*n* Entfernen des *n*-ten Verzeichnisses vom Stack; die Zählung beginnt bei 0.

-*n* Entfernen des *n*-ten Verzeichnisses vom unteren Teil des Stacks; die Zählung beginnt bei 0.

pushd *Verzeichnis*
pushd [*Optionen*]

pushd

Wenn kein Argument angegeben wird, werden die obersten beiden Verzeichnisse auf dem Stack vertauscht. Wenn ein Verzeichnis angegeben wird, wird es oben auf den Stack gelegt und dorthin verzweigt.

Optionen

+*n* Rotieren des Stacks, so daß das *n*-te Verzeichnis oben liegt; die Zählung beginnt bei 0.

-*n* Rotieren des Stacks, so daß das *n*-te Verzeichnis von unten gezählt oben liegt.

pwd	pwd Ausgeben des absoluten Pfadnamens des aktuellen Verzeichnisses. Wenn die Option -P angegeben wird, enthält der Pfadname keine symbolischen Links.

read

read [*Optionen*] *Variable1* [*Variable2* ...]

Einlesen einer Zeile von der Standardeingabe und Zuweisung jedes Wortes (was ein Wort ist, wird durch die Variable IFS definiert) an die entsprechende *Variable*. Alle übriggebliebenen Wörter werden der letzten Variablen zugewiesen. Wenn nur eine Variable angegeben ist, wird die gesamte Zeile dieser Variablen zugewiesen. Wenn überhaupt kein Variablenname angegeben wird, wird die gesamte Zeichenkette in die Umgebungsvariable REPLY eingelesen. Siehe das folgende Beispiel und die Beschreibung zum Befehl **case**. Der Ausführungsstatus ist 0, es sei denn, es wird ein Dateiendekennzeichen (EOF) eingelesen.

Optionen

-r „Roher" Modus; \ wird nicht als Zeilenfortsetzungszeichen interpretiert.

Beispiel

```
$ read Vorname Nachname Adresse
Susi Sorglos Bartolomaeusstr. 81

$ echo "$Nachname, $Vorname\n$Adresse"
Sorglos, Susi
Bartholomaeusstr. 81
```

readonly

readonly [*Optionen*] [*Variable1 Variable2* ...]

Verhindern, daß den angegebenen Shell-Variablen neue Werte zugewiesen werden. Die Variablen können gelesen, aber nicht überschrieben werden. In der **bash** kann die Schreibweise *Variable=Wert* verwendet werden, um einen nicht-überschreibbaren Wert zuzuweisen.

Optionen

-- Alle folgenden Zeichenketten werden als Argumente und nicht als Optionen interpretiert.

-f [*Variable(n)*]
Die Variablen *Variable(n)* werden auf Nur-Lesen gesetzt und können nicht geändert werden.

-p Ausgabe aller Nur-Lesen-Variablen (Default).

return

return [*n*]

Wird innerhalb einer Funktionsdefinition verwendet. Die Funktion wird mit einem Ausführungsstatus *n* oder dem Ausführungsstatus des zuletzt ausgeführten Befehls beendet.

select *Name* [*in Wortliste ;*]
do
  Befehle
done

select

Auswahl eines Wertes für die Variable *Name*, indem die Wörter in *Wortliste* auf dem Bildschirm ausgegeben werden und der Benutzer eine Eingabe machen muß. Die Eingabe des Benutzers wird in der Variablen REPLY, das ausgewählte Wort in *Name* abgelegt. Danach werden die *Befehle* so lange ausgeführt, bis ein **break** oder **return** ausgeführt wird.

set [*Optionen*] [*Arg1 Arg2* . . .]

set

Wenn keine Argumente angegeben werden, gibt **set** die Werte aller in der aktuellen Shell bekannten Variablen aus. Optionen können eingeschaltet (*–option*) oder ausgeschaltet (*+option*) werden. Die Optionen können auch beim Aufruf der Shell gesetzt werden. Die Argumente werden der Reihe nach **$1**, **$2** usw. zugewiesen.

bash

Optionen

- **-** Ausschalten von **-v** und **-x** und der Verarbeitung weiterer Optionen.
- **--** Wird als letzte Option verwendet; -- schaltet das Verarbeiten von Optionen ab, so daß Argumente, die mit einem – beginnen, nicht fälschlicherweise als Option interpretiert werden (beispielsweise kann **$1** auf –1 gesetzt werden). Wenn keine weiteren Argumente nach -- angegeben werden, werden die Positionsparameter gelöscht.
- **-a** Alle von nun an definierten oder neu zugewiesenen Variablen werden automatisch exportiert.
- **-b** Ausgabe des Status eines Hintergrund-Jobs, sobald dieser beendet ist (Default: wenn das nächste Shell-Prompt erscheint).
- **-d** Befehle werden nicht in die Hash-Liste aufgenommen, nachdem sie gefunden wurden.
- **-e** Beenden der Shell, wenn ein Befehl einen von 0 verschiedenen Ausführungsstatus hat.
- **-f** Platzhalterzeichen in Dateinamen werden nicht expandiert.
- **-h** Der Ort von Befehlen wird bereits bei ihrer Definition gemerkt.
- **-k** Die Zuweisung von Umgebungsvariablen (*Var*=*Wert*) findet unabhängig davon statt, wo sie auf der Kommandozeile stehen. Normalerweise müssen solche Zuweisungen vor dem Befehlsnamen stehen.
- **-l** Wenn in einem **for** eine Variable verwendet wird, die bereits einen Wert hat, wird der Wert dieser Variablen nach Beendigung des **for**-Befehls restauriert.
- **-m**
  Monitor-Modus. Job-Verwaltung wird eingeschaltet und Hintergrund-Jobs werden in einer gesonderten Prozeßgruppe ausgeführt. Dieser Modus wird normalerweise automatisch eingeschaltet.
- **-n** Befehle werden nur eingelesen, aber nicht ausgeführt. Dies ist zum Überprüfen von Shell-Skripten auf syntaktische Fehler nützlich.

→

-o [*m*]
Ausgeben der Shell-Modi oder Einschalten des Modus *m*. Viele Modi können auch mit anderen Optionen eingeschaltet werden. Die zur Verfügung stehenden Modi sind:

allexport	Wie -a.
braceexpand	Default. Expansion von Ausdrücken in geschweiften Klammern wird eingeschaltet.
emacs	Default. Emacs-artiges Editieren der Kommandozeile wird eingeschaltet.
errexit	Wie -e.
histexpand	Wie -H.
ignoreeof	Bei Eingabe von *EOF* die Shell nicht beenden. Um die Shell zu verlassen, muß **exit** eingegeben werden.
interactive-comments	Alle Wörter, die mit # beginnen, und alle darauffolgenden werden als Kommentar interpretiert.
monitor	Wie -m.
noclobber	Wie -C.
noexec	Wie -n.
noglob	Wie -f.
notify	Wie -b.
nounset	Wie -u.
physical	Wie -P.
posix	Erfüllen des POSIX-Standards.
privileged	Wie -p.
verbose	Wie -v.
vi	vi-artiges Editieren der Kommandozeile einschalten.
xtrace	Wie -x.

-p Starten als Superuser; *$HOME/.profile* nicht ausführen.

-t Beenden der Shell, nachdem ein Befehl ausgeführt wurde.

-u Variablen ohne Wert werden in Substitutionen als Fehler behandelt.

-v Jede Kommandozeile wird beim Einlesen wieder ausgegeben.

-x Anzeigen der Befehle und Argumente unter Voranstellung eines +, wenn die Befehle ausgeführt werden. Auf diese Art und Weise können Fehler in Shell-Skripten gesucht werden. (Wie **-o xtrace**.)

-C Wie **noclobber**

-H Default. !- und !!-Befehle werden eingeschaltet.

-P Ausgabe absoluter Pfadnamen bei **pwd**. Standardmäßig enthält die Ausgabe von **pwd** symbolische Links.

Beispiele

`set -- "$num" -20 -30`	*Setzen von* `$1` *auf den Wert von* num, *von* `$2` *auf –20 und von* `$3` *auf –30.*
`set -vx`	*Einlesen, Anzeigen und erneutes Anzeigen (mit Argumenten) jeder Kommandozeile.*
`set +x`	*Keine Kommandozeilen mit Argumenten mehr ausgeben.*
`set -o noclobber`	*Verhindern des Überschreibens von Dateien durch Umleitungsoperatoren.*

`set +o noclobber`	*Überschreiben von Dateien durch Umleitungsoperatoren wieder zulassen.*	set

shift [*n*] — shift

Weiterschieben der Positionsparameter ($2 wird $1 usw.). Wenn *n* angegeben wird, wird die Shift-Operation *n*-mal ausgeführt.

source *Datei* [*Argumente*] — source

Einlesen und Ausführen der Befehle in *Datei*. *Datei* muß nicht ausführbar sein, aber in einem Verzeichnis stehen, das in PATH enthalten ist.

suspend [-f] — suspend

Wie **CTRL-Z**. **suspend** wirkt aber nicht auf einen Hintergrundjob, sondern auf die Shell selbst. Wird oft benutzt, um den **su**-Befehl abzubrechen.

Optionen

-f Anhalten erzwingen, auch wenn die Shell eine Anmelde-Shell ist.

bash

test *Bedingung*
oder
[*Bedingung*] — test

Auswertung einer Bedingung *Bedingung* und Rückgabe eines Ausführungsstatus von 0, wenn die Bedingung wahr ist. Anstelle des Wortes **test** kann auch [ ] verwendet werden. Die Bedingungen sind wahr, wenn die folgende Beschreibung wahr ist:

Datei-basierte Bedingungen

-b *Datei*	*Datei* existiert und ist eine Block-Geräte-Datei (block special file).
-c *Datei*	*Datei* existiert und ist eine Zeichen-Geräte-Datei (character special file).
-d *Datei*	*Datei* existiert und ist ein Verzeichnis.
-e *Datei*	*Datei* existiert.
-f *Datei*	*Datei* existiert und ist eine reguläre Datei.
-g *Datei*	*Datei* existiert, und das SGID-Bit ist gesetzt.
-k *Datei*	*Datei* existiert, und das Sticky-Bit ist gesetzt.
-p *Datei*	*Datei* existiert und ist eine benannte Pipe (FIFO).
-r *Datei*	*Datei* existiert und ist lesbar.
-s *Datei*	*Datei* existiert und ist größer als 0.
-t [*n*]	Der offene Datei-Deskriptor *n* wird mit einem Terminal assoziiert; der Default für *n* ist 1.
-u *Datei*	*Datei* existiert, und das SUID-Bit ist gesetzt.
-w *Datei*	*Datei* existiert und ist schreibbar.

→

-x *Datei*	*Datei* existiert und ist ausführbar.
-G *Datei*	*Datei* existiert, und die Gruppe des Besitzers ist gleich der effektiven Gruppen-ID des Prozesses.
-L *Datei*	*Datei* existiert und ist ein symbolischer Link.
-O *Datei*	*Datei* existiert, und die ID des Besitzers ist gleich der effektiven Benutzer-ID des Prozesses.
-S *Datei*	*Datei* existiert und ist ein Socket.
f1 -ef *f2*	Die Dateien *f1* und *f2* sind gelinkt (bezeichnen dieselbe Datei).
f1 -nt *f2*	Die Datei *f1* ist neuer als die Datei *f2*.
f1 -ot *f2*	Die Datei *f1* ist älter als die Datei *f2*.

String-basierte Bedingungen

-n *s1*	Die Zeichenkette *s1* ist länger als null.
-z *s1*	Die Zeichenkette *s1* hat die Länge null.
s1 = *s2*	Die Zeichenketten *s1* und *s2* sind identisch.
s1 != *s2*	Die Zeichenketten *s1* und *s2* sind *nicht* identisch.
String	Die Zeichenkette *String* ist nicht null.

Ganzzahl-basierte Bedingungen

n1 -eq *n2*	*n1* ist gleich *n2*.
n1 -ge *n2*	*n1* ist größer als oder gleich *n2*.
n1 -gt *n2*	*n1* ist größer als *n2*.
n1 -le *n2*	*n1* ist kleiner als oder gleich *n2*.
n1 -lt *n2*	*n1* ist kleiner als *n2*.
n1 -ne *n2*	*n1* ist ungleich *n2*.

Verknüpfungen von Bedingungen

! *Bedingung* Wahr, wenn *Bedingung* falsch ist.

Bedingung1 -a *Bedingung2*
Wahr, wenn beide Bedingungen wahr sind.

Bedingung1 -o *Bedingung2*
Wahr, wenn mindestens eine der beiden Bedingungen wahr ist.

Beispiele

Jedes der folgenden Beispiele zeigt die erste Zeile eines Befehls, der eine Bedingungsüberprüfung verwenden könnte:

`while test $# -gt 0`	*Solange noch Argumente vorhanden sind . . .*
`while [ -n "$1" ]`	*Solange das erste Argument nicht leer ist . . .*
`if [ $zaehler -lt 10 ]`	*Wenn* `$zaehler` *kleiner als 10 ist . . .*
`if [ -d RCS ]`	*Wenn das Verzeichnis RCS existiert . . .*
`if [ "$antwort" != "y" ]`	*Wenn die Antwort nicht* y *ist . . .*
`if [ ! -r "$1" -o ! -f "$1" ]`	*Wenn das erste Argument entweder eine nichtlesbare Datei oder keine reguläre Datei ist . . .*

times

Ausgabe der aufgelaufenen Prozeßlaufzeiten getrennt nach Benutzer und System.

times

trap [–l] [[*Befehle*] *Signale*]

trap

Ausführen der Befehle *Befehle*, wenn der Prozeß eines der Signale in *Signale* erhält. Oft beim **trap**-Befehl verwendete Signale sind 0, 1, 2 und 15. Wenn mehrere Befehle aufgerufen werden sollen, sollten diese als Gruppe zusammengefaßt und innerhalb der Gruppe mit Semikola voneinander getrennt werden. Wenn *Befehle* die leere Zeichenkette ist (d.h. **trap** "" *Signale*), ignoriert die Shell die Signale. Wenn *Befehle* völlig weggelassen wird, wird die Behandlung der angegebenen Signale auf die Default-Werte zurückgesetzt. Wenn sowohl *Befehle* als auch *Signale* weggelassen werden, wird der aktuelle Status der Signal-Behandlung ausgegeben. Siehe die Beispiele am Ende dieses Eintrags und unter **exec**.

Option

-l   Ausgeben der Signale.

Signale

Die folgende Aufstellung enthält die Signale und die Zustände, die sie auslösen.

0	Beenden der Shell (üblicherweise bei Beenden der Abarbeitung eines Shell-Skriptes)
1	Beenden der Verbindung (üblicherweise Abmelden)
2	Interrupt (üblicherweise **CTRL-C**)
3	Beenden
4	Illegaler Maschinenbefehl
5	Trace trap
6	Abbruch
7	Nicht benutzt
8	Fließkomma-Ausnahmebedingung
9	Erzwungener Abbruch
10	Benutzer-definiert
11	Zugriff auf ungültigen Speicherbereich („segmentation fault")
12	Benutzer-definiert
13	Schreiben in eine Pipe, ohne daß ein Prozeß aus dieser liest
14	Ablaufen eines Timers
15	Abbruch eines Programms (üblicherweise via **kill**)
16	Stackfehler im Coprozessor
17	Beenden eines Kind-Prozesses
18	Fortsetzen (wenn der Prozeß angehalten war)
19	Anhalten des Prozesses

→

trap ←

20	Prozeß wurde vom Terminal aus (meist mit **CTRL-Z**) angehalten
21	Hintergrund-Prozeß will von einem Terminal lesen
22	Hintergrund-Prozeß will auf ein Terminal ausgeben
23	E/A-Fehler
24	CPU-Zeit-Beschränkung erreicht
25	Dateigrößen-Beschränkung erreicht
27	Ein mit setitimer(2) gestarteter Profile-Timer ist abgelaufen
28	Veränderung der Fenstergröße

Beispiele

```
trap "" 2        Ignorieren des Signals 2 (Interrupts)
trap 2           Interrupts werden wieder befolgt
```

Entfernen der Datei **$tmp** und Beenden des Prozesses, wenn das Shell-Skript endet, der Benutzer sich abmeldet, auf **CTRL-C** drückt oder **kill** aufruft:

```
trap "rm -f $tmp; exit" 0 1 2 15
```

type

type [*Optionen*] *Befehle*

Anzeigen der absoluten Pfadnamen von Befehlen sowie der Information, ob sich diese Befehle in der Hash-Tabelle der aktuellen Shell befinden.

- `--` Alle folgenden Zeichenketten sind keine Optionen, sondern immer Argumente.
- **-a, -all** Alle Vorkommen von *Befehl* werden ausgegeben, nicht nur das, welches aufgerufen werden würde.
- **-p, -path** Anzeigen des Eintrags von *Befehl* in der Hash-Tabelle. Dieser Wert kann sich vom ersten Auftreten des Befehls in PATH unterscheiden.
- **-t, -type** Anzeigen, ob *Befehl* ein Aliasname, ein Schlüsselwort, eine Funktion, ein eingebauter Befehl oder eine Datei ist.

Beispiel

```
$ type mv read
mv is /bin/mv
read is a shell built-in
```

typeset

typeset

Siehe **declare**.

ulimit

ulimit [*Optionen*] [*n*]

Ausgeben oder Setzen (wenn *n* angegeben ist) von Ressourcen-Beschränkungen. Eine Ressourcen-Beschränkung kann entweder hart (-H) oder weich (-S) sein. Standardmäßig setzt **ulimit** beide Beschränkungen und gibt die weiche aus. Die Optionen bestimmen, um welche Ressource es sich handelt.

ulimit

Optionen

- `--` Alle folgenden Zeichenketten werden als Argumente und nicht als Optionen behandelt.
- `-a` Ausgabe aller aktuell gesetzten Beschränkungen.
- `-H` Harte Ressourcen-Beschränkung.
- `-S` Weiche Ressourcen-Beschränkung.

Setzbare Beschränkungen

- `-c` Core-Dateien.
- `-d` Größe des Datensegments von Prozessen.
- `-f` Größe von Dateien, die aus der Shell heraus angelegt werden.
- `-m` Größe des Residuums eines Prozesses.
- `-n` Anzahl der Datei-Deskriptoren. Kann auf vielen Systemen nicht gesetzt werden.
- `-p` Größe von Pipes in 512-Byte-Blöcken.
- `-s` Größe des Stacks.
- `-t` CPU-Zeit in Sekunden.
- `-u` Anzahl der Prozesse pro Benutzer.
- `-v` Von der Shell verwendeter virtueller Speicher.

bash

---

umask

umask [*nnn*]
umask [*-S*]

Anzeigen oder Setzen der beim Erzeugen von Dateien verwendeten Rechte-Maske. *nnn* wird als Oktalzahl interpretiert. Die Rechte-Maske bestimmt, welche Dateizugriffsrechte bei neu angelegten Dateien gesetzt werden (beispielsweise führt **umask 002** zu den Dateizugriffsrechten **rw-rw-r--**).

Option

- `-S` Der Wert von umask wird nicht oktal sondern symbolisch ausgegeben.

---

unalias

unalias [*–a*] *Namen*

Entfernt *Namen* von der Liste der Aliasnamen. Siehe auch **alias**.

Option

- `-a` Entfernt alle Aliasnamen.

---

unset

unset [*Optionen*] *Namen*

Löscht die Funktions- oder Variablendefinitionen, die in *Namen* angegeben sind.

Optionen

- `-f` *Name* wird auf jeden Fall als Funktion interpretiert.
- `-v` *Name* wird auf jeden Fall als Variable interpretiert (default).

until	until *Test-Befehle* do *Befehle* done Ausführen der *Test-Befehle* (üblicherweise **test** oder []). Falls der Ausführungsstatus nicht null ist (der Test also gescheitert ist), werden die *Befehle* ausgeführt und die ganze Prozedur wiederholt.
wait	wait [*ID*] Hält die Programmausführung an, bis alle Hintergrund-Jobs vollständig ausgeführt sind (als Ausführungsstatus wird 0 zurückgegeben) oder bis der durch seine Prozeß- oder Job-*ID* angegebene Hintergrund-Prozeß beendet ist (als Ausführungsstatus wird der Ausführungsstatus des Prozesses zurückgegeben). Beachten Sie, daß die Shell-Variable $! die Prozeß-ID des jüngsten Hintergrund-Prozesses enthält. Wenn die Job-Verwaltung abgeschaltet ist, kann *ID* nur eine Prozeß-ID bezeichnen. Siehe auch „Job-Verwaltung". ***Beispiel*** `wait $!` *Warten auf die Beendigung des letzten Hintergrund-Prozesses.*
while	while *Test-Befehle* do *Befehle* done Ausführen der *Test-Befehle* (üblicherweise **test** oder []); wenn der Ausführungsstatus 0 ist (der Test also erfolgreich war), werden die *Befehle* ausgeführt und die gesamte Prozedur wiederholt.

Job-Verwaltung

Die Job-Verwaltung ermöglicht es Ihnen, im Vordergrund laufende Jobs in den Hintergrund zu setzen, im Hintergrund laufende Jobs in den Vordergrund zu holen oder Jobs vorübergehend anzuhalten. Standardmäßig ist die Job-Verwaltung eingeschaltet. Wenn die Job-Verwaltung abgeschaltet wurde, kann sie mit einem der folgenden Befehle wieder eingeschaltet werden:

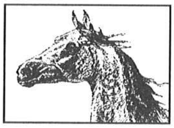
```
bash -m -i
set -m
set -o monitor
```

Viele Befehle der Job-Verwaltung erwarten eine *JobID* als Argument. Dieses Argument kann folgendermaßen angegeben werden:

%*n*	Job-Nummer *n*
%*s*	Job, dessen Kommandozeile mit der Zeichenkette *s* anfängt
%?*s*	Job, dessen Kommandozeile die Zeichenkette *s* enthält
%%	Aktueller Job
%+	Aktueller Job (wie oben)
%-	Vorheriger Job

Es gibt in der **bash** die folgenden Befehle zur Job-Verwaltung. Mehr Information zu diesen Befehlen finden Sie unter „Eingebaute Befehle“ weiter vorn in diesem Kapitel.

bg	Einen Job in den Hintergrund stellen.
fg	Einen Job in den Vordergrund holen.
jobs	Aktive Jobs ausgeben.
kill	Beenden eines Jobs.
stop	Einen Hintergrund-Job vorübergehend anhalten.
stty tostop	Hintergrund-Jobs anhalten, wenn sie Ausgaben auf das Terminal ausgeben wollen.
wait	Warten, daß der Hintergrund-Job endet.
CTRL-Z	Einen Vordergrund-Job vorübergehend anhalten. Danach wird üblicherweise **bg** oder **fg** verwendet. (Manche Terminals verwenden andere Zeichen als **CTRL-Z**, um das vorübergehende Anhalten zu bewirken.



KAPITEL 4

csh und tcsh

Dieses Kapitel beschreibt die C-Shell und deren erweiterte Version, **tcsh**. Die C-Shell erhielt ihren Namen, weil viele ihrer Konstrukte zum Programmieren denen der Programmiersprache C ähnlich sind. Sie finden in diesem Kapitel die folgenden Abschnitte:

- Funktionsübersicht
- Aufruf der Shell
- Syntax
- Variablen
- Ausdrücke
- Befehls-History
- Bearbeiten der Kommandozeile
- Eingebaute Befehle
- Job-Verwaltung

Funktionsübersicht

Die C-Shell hat unter anderem die folgenden Funktionen:

- Ein-/Ausgabe-Umleitung
- Platzhalterzeichen (Metazeichen) zur Abkürzung von Dateinamen
- Shell-Variablen, um die Arbeitsumgebung einzurichten
- Integer-Arithmetik
- Zugriff auf früher eingegebene Befehle (Befehls-History)
- Abkürzen von Befehlen (Verwenden von Aliasnamen)
- Eingebaute Befehle, um Shell-Skripten zu schreiben
- Job-Manipulation

Die **tcsh** enthält alle bereits erwähnten Funktionen und verfügt außerdem über die folgenden Erweiterungen:

- Editieren der Kommandozeile und Editor-Befehle
- Vervollständigung von Wörtern (mit der Tabulatortaste)
- Rechtschreibprüfung
- Erweiterte Befehle zur Bearbeitung der History
- Erweiterte Manipulation von Verzeichnissen (**cd**, **pushd**, **popd** und **dirs**)
- Ausführen von Funktionen bei bestimmten Ereignissen oder nach bestimmten Zeitabschnitten, z.B. Abmelden von einem Terminal, nachdem eine gewisse Zeit nichts eingegeben wurde, und verzögertes Ausführen von Befehlen
- Neue eingebaute Befehle: **hup**, **ls-F**, **newgrp**, **printenv**, **which** und **where**
- Neue Shell-Variablen (**gid**, **loginsh**, **oid**, **shlvl**, **tcsh**, **tty**, **uid** und **version**) und Umgebungsvariablen (**HOST**, **REMOTEHOST**, **VENDOR**, **OSTYPE** und **MACHTYPE**)
- Neue Formatiersequenzen für die PROMPT-Variable und zwei neue Eingabeaufforderungen (für Schleifen und die Rechtschreibprüfung)
- Variablen, die nur gelesen werden können

Aufrufen der Shell

Der Befehlsinterpreter kann folgendermaßen aufgerufen werden:

> **csh** [*Optionen*] [*Argumente*]
> **tcsh** [*Optionen*] [*Argumente*]

csh und **tcsh** verwenden eine Syntax, die der Programmiersprache C ähnelt, und führen Befehle aus, die an einem Terminal eingegeben oder aus einer Datei gelesen wurden. Die Optionen **-n**, **-v** und **-x** sind nützlich zum Debuggen von Shell-Skripten.

Optionen

-b Die folgenden Kommandozeilenoptionen werden als Optionen eines angegebenen Befehls interpretiert und nicht als Optionen der Shell selbst.

-c Ausführen von Befehlen in der Datei, deren Name dieser Option folgt.

-e Beenden der Shell, wenn ein Fehler auftritt.

-f Schnellstart; die Dateien *.cshrc* und *.login* werden nicht ausgeführt.

-i Interaktive Shell aufrufen (die Shell fragt nach einer Eingabe).

-n Befehle syntaktisch analysieren, aber nicht ausführen.

-s Befehle von der Standardeingabe lesen.

-t Beenden der Shell, nachdem der erste Befehl ausgeführt wurde.

-v Befehle vor ihrer Ausführung anzeigen; nur History-Ersetzungen ausführen (also keine Dateinamen-, Variablen-, Befehlsersetzungen etc.). Kann auch durch Setzen der Variable **verbose** erreicht werden.

-V Wie **-v**, zusätzlich Anzeigen der Datei *.cshrc*.

-x Befehle vor ihrer Ausführung anzeigen. Alle Ersetzungen werden durchgeführt. Kann auch durch Setzen der Variable **echo** erreicht werden.

-X Wie **-x**, zusätzlich Anzeigen der Datei *.cshrc*.

Argumente

Argumente werden in der Reihenfolge ihres Auftretens den Positionsparametern **$1**, **$2** etc. zugewiesen. Wenn das erste Argument eine ausführbare Datei ist, werden die darin enthaltenen Befehle gelesen und die verbleibenden Argumente den Positionsparametern zugewiesen.

Syntax

Dieser Abschnitt beschreibt die vielen Symbole, die in der **csh** und **tcsh** eine Sonderbedeutung haben. Er ist in die folgenden Unterabschnitte gegliedert:

- Besondere Dateien
- Metazeichen für Dateinamen
- Quoten
- Schreibweisen von Befehlen
- Schreibweisen von Umleitungen

Besondere Dateien

~/.cshrc	Wird jedesmal beim Starten einer Shell ausgeführt
~/.login	Wird beim Anmelden nach *.cshrc* ausgeführt
~/.logout	Wird beim Abmelden ausgeführt
/etc/passwd	Quelle für Heimatverzeichnisse in ~*name*-Abkürzungen

Metazeichen in Dateinamen

`*`	Paßt auf jede Zeichenkette.
`?`	Paßt auf ein einzelnes beliebiges Zeichen.
`[`*abc*`...]`	Paßt auf eines der Zeichen in den eckigen Klammern. Mit Bindestrichen können Bereiche angegeben werden (z.B. a-z, A-Z, 0-9).
`{`*abc*`,`*xxx*`,...}`	Paßt auf einen der durch Kommata getrennten Strings in den geschweiften Klammern.
`~`	Heimatverzeichnis des Benutzers.
`~`*Name*	Heimatverzeichnis des Benutzers *Name*.

Beispiele

`% ls neu*`	*Paßt auf* `neu` *und* `neu.1`
`% cat kap?`	*Paßt auf* `kap9`*, aber nicht auf* `kap10`
`% vi [D-R]*`	*Paßt auf alle Dateien, die mit einem zwischen D und R (einschließlich) liegenden Dateinamen beginnen*
`% ls {kap,app}?`	*Paßt auf* `kap1, kap2, app1, app2`
`% cd ~tom`	*Wechsel in das Heimatverzeichnis des Benutzers* `tom`

Quoten

Quoten bezeichnet das Abschalten der Sonderbedeutung eines Zeichens, so daß dieses Zeichen wörtlich verwendet werden kann. In der C-Shell haben die folgenden Zeichen eine Sonderbedeutung:

`;`	Trennzeichen zwischen zwei Befehlen
`&`	Ausführung im Hintergrund
`( )`	Gruppieren von Befehlen
`|`	Pipe
`* ? [ ] ~`	Metazeichen für Dateinamen
`{ }`	Expansionszeichen für Strings (müssen normalerweise nicht gequotet werden)
`> < & !`	Symbole für Umleitungen
`! ^`	History-Ersetzung, Schnellersetzung
`" ' \`	Werden zum Quoten anderer Zeichen verwendet
`` ` ``	Befehls-Ersetzung
`$`	Variablen-Ersetzung
Zeilenwechsel **Leerzeichen** **Tabulatorzeichen**	Trennzeichen zwischen Wörtern

Die folgenden Zeichen können zum Quoten verwendet werden:

`" "` Alles zwischen `"` und `"` wird wortwörtlich interpretiert mit Ausnahme der folgenden Zeichen, die ihre Sonderbedeutung behalten:

`$`	Variablen-Ersetzung wird ausgeführt.
`` ` ``	Befehls-Ersetzung wird ausgeführt.
`"`	Ende des gequoteten Bereichs.
`\`	Nächstes Zeichen quoten.
`!`	History-Zeichen
Zeilenwechsel	Das Zeilenwechselzeichen

`' '` Alles zwischen `'` und `'` wird wortwörtlich genommen. Ausgenommen davon sind nur ! (History), ein weiteres `'` und der Zeilenwechsel.

`\` Das dem Backslash folgende Zeichen wird wörtlich genommen. Der Backslash wird innerhalb von " " verwendet, um ", $ und ` zu quoten. Wird oft verwendet, um einen weiteren Backslash, Leerzeichen oder Zeilenwechsel zu quoten. Der Backslash muß immer verwendet werden, um das History-Zeichen (normalerweise !) zu quoten.

Beispiele

```
% echo 'Einfache Anführungszeichen "schützen" doppelte Anführungszeichen'
Einfache Anführungszeichen "schützen" doppelte Anführungszeichen

% echo "Ist das nicht etwas \"Besonderes\"?"
Ist das nicht etwas "Besonderes"?
```

```
% echo "Sie haben `ls|wc -l` Dateien in `pwd`"
Sie haben 43 Dateien in /home/kalle

% echo "Der Wert von \$x ist $x"
Der Wert von $x ist 100
```

Schreibweisen von Befehlen

Befehl &	Ausführen von *Befehl* im Hintergrund.
Befehl1 ; *Befehl2*	Folge von Befehlen; Ausführen mehrerer *Befehle* auf ein und derselben Zeile.
(*Befehl1* ; *Befehl2*)	Sub-Shell; *Befehl1* und *Befehl2* werden als Befehlsgruppe betrachtet.
Befehl1 | *Befehl2*	Pipe; die Ausgabe von *Befehl1* ist die Eingabe von *Befehl2*.
Befehl1 `*Befehl2*`	Befehlsersetzung, die Ausgabe von *Befehl2* wird als Argument für *Befehl1* verwendet.
Befehl1 && *Befehl2*	Logisches Und; *Befehl2* wird ausgeführt, falls die Ausführung von *Befehl1* erfolgreich war.
Befehl1 || *Befehl2*	Logisches Oder; *Befehl2* wird ausgeführt, falls die Ausführung von *Befehl1* nicht erfolgreich war.

Beispiele

```
% nroff Datei &                      Im Hintergrund formatieren
% cd; ls                             Nacheinander ausführen
% (date; who; pwd) > logdatei        Alle Ausgaben umleiten
% sort Datei | pr -3 | lp            Datei sortieren, zum Druck formatieren und drucken
% vi `grep -l ifdef *.c`             Von grep gefundene Dateien editieren
% egrep '(yes|no)' `cat Liste`       Liste zu durchsuchender Dateien angeben
% grep XX Datei || lp Datei          Datei drucken, wenn sie das Suchmuster enthält
% grep XX Datei && echo XX nicht gefunden  Fehlermeldung ausgeben, wenn XX
                                           nicht gefunden wird
```

Schreibweisen von Umleitungen

Datei-Deskriptor	Name	Übliche Abkürzung	Typischer Default
0	Standardeingabe	stdin	Tastatur
1	Standardausgabe	stdout	Bildschirm
2	Standardfehler	stderr	Bildschirm

Die normale Ein- oder Ausgabequelle kann folgendermaßen geändert werden:

Einfache Umleitungen

Befehl > *Datei*	Ausgabe von *Befehl* nach *Datei* schreiben (alte Datei wird überschrieben).
Befehl >! *Datei*	Wie oben, auch wenn **noclobber** gesetzt ist.
Befehl >> *Datei*	Ausgabe von *Befehl* an *Datei* (anhängen).
Befehl >>! *Datei*	Wie oben, auch wenn **noclobber** gesetzt ist.

Befehl < *Datei*	Eingabe für *Befehl* aus *Datei* holen.
Befehl << *Text*	Von der Standardeingabe lesen, bis eine Zeile kommt, die mit *Text* identisch ist (*Text* kann auch in einer Shell-Variable stehen). Die Eingabe wird normalerweise am Bildschirm eingegeben oder stammt aus einem Shell-Skript. Diese Schreibweise wird meist in **cat**-, **echo**-, **ex**- und **sed**-Befehlen verwendet. Wenn *Text* in Anführungszeichen steht, wird auf die Eingabe keine Variablen-, Befehls- und andere Ersetzung angewendet.

Mehrfache Umleitung

Befehl >& *Datei*	Standardausgabe und Standardfehler nach *Datei* schreiben.
Befehl >&! *Datei*	Wie oben, auch wenn **noclobber** gesetzt ist.
Befehl >>& *Datei*	Standardausgabe und Standardfehler an *Datei* anhängen.
Befehl >>&! *Datei*	Wie oben, auch wenn **noclobber** gesetzt ist.
Befehl1 |& *Befehl2*	Standardausgabe und Standardfehler von *Befehl1* als Eingabe von *Befehl2* verwenden.
(*Befehl* > *f1*) >& *f2*	Standardausgabe nach *f1*, Standardfehler nach *f2* schreiben.
Befehl | tee *Dateien*	Ausgabe von *Befehl* auf die Standardausgabe (üblicherweise der Bildschirm) und nach *Dateien* schreiben. (Siehe das Beispiel in Kapitel 1 unter **tee**.)

Beispiele

```
% cat teil1 > buch                       teil1 nach buch kopieren
% cat teil2 teil3 >> buch                Teile 2 und 3 an dieselbe Datei wie teil1 anhängen
% mail elke < report                     Eingabe für die Nachricht aus report nehmen
% cc calc.c >& error_out                 Alle Meldungen, auch Fehler, abspeichern
% cc newcalc.c >&! error_out             Alte Datei überschreiben
% grep UNIX ch* |& pr                    Alle Meldungen, auch Fehler, durch die Pipe schicken
% (find / -print > Dateilist) >& no_access Fehlermeldungen und Dateiliste
                                         voneinander trennen
% sed 's/^/XX /' << "END_ARCHIVE"        Text wird direkt hinter dem Befehl angegeben
Auf diese Art und Weise wird oft Text zur
Weitergabe in einem Shell-Skript verpackt. Normalerweise
wird sed aus einem Shell-Skript und nicht von der
Kommandozeile aufgerufen
"END_ARCHIVE"
```

Variablen

Dieser Abschnitt enthält die folgenden Themen:

- Variablen-Ersetzung
- Modifikatoren für Variablen
- Vordefinierte Shell-Variablen
- Beispielhafte *.cshrc*-Datei
- Umgebungsvariablen

Variablen-Ersetzung

In den folgenden Ersetzungen können die geschweiften Klammern ({ }) weggelassen werden, es sei denn, sie trennen einen Variablennamen von den folgenden Zeichen, die sonst zum Variablennamen gehören würden.

`${`*`Var`*`}`	Der Wert der Variablen *Var*.
`${`*`Var[i]`*`}`	Auswahl einzelner Wörter an Position *i* in *Var*. *i* kann eine einzelne Zahl, ein Bereich *m–n*, ein Bereich *-n* (das fehlende *m* impliziert 1), ein Bereich *m-* (das fehlende *n* impliziert alle folgenden Wörter) oder * (alle Wörter) sein, außerdem kann *i* wiederum eine Variable sein, deren Wert einer der genannten Werte ist.
`${#`*`Var`*`}`	Die Anzahl von Wörtern in *Var*.
`${#argv}`	Anzahl der Argumente.
`$0`	Name des Programms.
`${#argv[`*`n`*`]}`	Die Argumente der Kommandozeile (Positionsparameter), $1 \leq n \leq 9$.
`${`*`n`*`}`	Wie **${argv[*n*]}**.
`${#argv[*]}`	Alle Argumente der Kommandozeile.
`$*`	Wie **$argv[*]**.
`$argv[$#argv]`	Das letzte Argument.
`${?`*`var`*`}`	Gibt 1 zurück, wenn *Var* gesetzt ist, ansonsten 0.
`$$`	Prozeßnummer der aktuellen Shell; kann gut als Teil des Dateinamens von temporären Dateien verwendet werden, um eindeutige Namen zu gewährleisten.
`$?0`	1 ausgeben, wenn der Eingabe-Dateiname bekannt ist, ansonsten 0.

csh und tcsh

Beispiele

Sortieren des dritten bis letzten Arguments; die Ausgabe wird in einer Datei mit einem eindeutigen Namen gespeichert:

```
sort $argv[3-] > tmp.$$
```

Befehle in *.cshrc* nur ausführen, wenn es sich um eine interaktive Shell handelt (in diesem Fall ist die **prompt**-Variable gesetzt).

```
if ($?prompt) then
    set commands
    alias commands
    etc.
endif
```

Modifikatoren für Variablen

Außer $?*var*, $$ und $?0 können allen Variablen-Ersetzungen des vorigen Abschnitts die untenstehenden Modifikatoren folgen. Wenn geschweifte Klammern verwendet werden, müssen die Modifikatoren innerhalb der geschweiften Klammern stehen.

- :r Zurückgeben der Wurzel der Variablen (alles vor dem letzten Punkt).
- :e Zurückgeben der Extension der Variablen (alles nach dem letzten Punkt).

:h Zurückgeben des Kopfes der Variablen (der Verzeichnisanteil).

:t Zurückgeben des Schwanzes der Variablen (alles nach dem letzten Schrägstrich).

:gr Alle Wurzeln zurückgeben.

:ge Alle Extensionen zurückgeben.

:gh Alle Köpfe zurückgeben.

:gt Alle Schwänze zurückgeben.

:q Quoten einer Variablen, die eine Wortliste enthält, wobei die einzelnen Elemente voneinander getrennt bleiben. Dies ist nützlich, wenn die Variable Metazeichen enthält, die nicht expandiert werden sollen.

:x Quoten eines Musters, das zu einer Wortliste expandiert wird.

Beispiele mit Modifikatoren für Pfadnamen

Die folgende Tabelle zeigt die Verwendung von Modifikatoren für Dateinamen anhand der folgenden Variable:

```
set aa=(/progs/num.c /buch/kap.ps)
```

Variablenteil	Spezifikation	Ergebnis
Ganze Variable	`echo $aa`	/progs/num.c /buch/kap.ps
Zweite Wurzel	`echo $aa[2]:r`	/buch/kap
Zweiter Kopf	`echo $aa[2]:h`	/buch
Zweiter Schwanz	`echo $aa[2]:t`	kap.ps
Zweite Extension	`echo $aa[2]:e`	ps
Wurzel	`echo $aa:r`	/progs/num /buch/kap.ps
Globale Wurzel	`echo $aa:gr`	/progs/num /buch/kap
Kopf	`echo $aa:h`	/progs /buch/kap.ps
Globaler Kopf	`echo $aa:gh`	/progs /buch
Schwanz	`echo $aa:t`	num.c /buch/kap.ps
Globaler Schwanz	`echo $aa:gt`	num.c kap.ps
Extension	`echo $aa:e`	c /buch/kap.ps
Globale Extension	`echo $aa:ge`	c ps

Beispiele für quotende Modifikatoren

Wenn sie nicht gequotet werden, expandiert die Shell Variablen zu den Dateinamen des aktuellen Verzeichnisses.

```
% set a="[a-z]*" A="[A-Z]*"
% echo "$a" "$A"
[a-z]* [A-Z]*

% echo $a $A
at cc m4 Buch Doc

% echo $a:x $A
[a-z]* Buch Doc

% set d=($a:q $A:q)
% echo $d
at cc m4 Buch Doc
```

```
% echo $d:q
[a-z]* [A-Z]*

% echo $d[1] +++ $d[2]
at cc m4 +++ Buch Doc

% echo $d[1]:q
[a-z]*
```

Vordefinierte Shell-Variablen

Variablen können auf zweierlei Art und Weise gesetzt werden: Zum einen durch Wertzuweisung:

`set Var=Wert`

zum anderen durch einfaches Setzen der Variablen:

`set Var`

In der folgenden Liste werden Variablen, denen ein Wert zugewiesen werden kann, mit einem Gleichheitszeichen gekennzeichnet. Dahinter folgt der Typ von Werten, den die Variable akzeptiert, einschließlich einer Beschreibung des Typs. (Beachten Sie aber, daß Variablen wie **argv**, **cwd** und **status** nie Werte zugewiesen werden.) Bei Variablen, die lediglich gesetzt und zurückgesetzt werden, wird in der Tabelle angegeben, was sie bedeuten, wenn sie gesetzt sind. Die **tcsh** setzt (und aktualisiert in manchen Fällen) die Variablen **addsuffix**, **argv**, **autologout**, **cwd**, **dirstack**, **echo-style**, **edit**, **gid**, **home**, **loginsh**, **logout**, **oid**, **owd**, **path**, **prompt**, **prompt2**, **prompt3**, **shell**, **shlvl**, **status**, **tcsh**, **term tty**, **uid**, **user** und **version**. Variablen in kursiven Buchstaben sind **tcsh**-spezifisch.

csh und tcsh

addsuffix	Anhängen eines / und eines Leerzeichens an Dateien bei der Tab-Vervollständigung, wenn ein Name genau paßt.
ampm	Alle Zeitangaben in zwölfstündigem Format machen.
argv=(*Args*)	Liste der Argumente des aktuellen Befehls, der Default ist `()`.
autocorrect	Rechtschreibung vor der Vervollständigung von Befehlen überprüfen.
autoexpand	History-Expansion (z.B. !-Referenzen) während der Befehls-Vervollständigung durchführen.
autolist[=ambiguous]	Mögliche Vervollständigungen angeben, wenn die richtige noch nicht bestimmt werden kann. Wenn **ambiguous** angegeben wird, werden die möglichen Vervollständigungen nur ausgegeben, wenn dazu keine neuen Zeichen hinzugefügt werden müssen.
autologout=logout-Minuten [locking-Minuten]	Abmelden, wenn nach *logout-Minuten* keine Eingabe erfolgt. Wenn nach *locking-Minuten* keine Eingabe erfolgt, wird das Terminal blockiert; zum Weiterarbeiten muß ein Paßwort eingegeben werden.
backslash_quote	Backslashes dürfen immer \, ' und " quoten.
cdpath=*Dirs*	Liste der alternativen Verzeichnisse, die bei **cd**, **popd** und **pushd** durchsucht werden.

complete=enhance	Wenn diese Variable auf **enhance** gesetzt ist, wird beim Vervollständigen keine Groß-/Kleinschreibung berücksichtigt, ., – und _ werden als Worttrennzeichen und _ und – als gleich betrachtet.
correct={cmd|complete|all}	Wenn der Wert **cmd** ist, werden Befehle auf Rechtschreibung überprüft; bei **complete** werden Befehle vervollständigt, und bei **all** wird die gesamte Kommandozeile auf Rechtschreibung überprüft.
cwd=*Dir*	Voller Pfadname des aktuellen Verzeichnisses.
dirsfile=Datei	History-Datei, die von **dirs -S** und **dirs -L** verwendet wird. Der Default ist *~/.cshdirs*.
dirstack	Verzeichnis-Stack im Array-Format. *dirstack[0]* ist immer gleich wie **cwd**. Alle anderen Elemente können von Hand verändert werden.
dunique	Sicherstellen, daß jedes Verzeichnis nur einmal im Stack existiert.
echo	Ausgeben jeder Kommandozeile vor ihrer Ausführung; wie **csh -x**.
echo_style={bsd|sysv|both|none}	Keinen Zeilenwechsel ausgeben, wenn die Option **-n** verwendet wird (**bsd**) **|** gequotete Zeichen parsen (**sysv**) **|** beides tun (**both**) **|** keines von beidem tun (**none**).
edit	Kommandozeileneditor einschalten.
ellipsis	Wird mit der **prompt**-Variable verwendet. Ausgelassene Verzeichnisse werden durch ... dargestellt.
fignore=Zeichen	Liste von Dateisuffixen, die bei der Vervollständigung ignoriert werden (siehe **filec**).
gid	Gruppen-ID des Benutzers.
histchars=*ab*	Ein aus zwei Zeichen bestehender String, der die Zeichen festlegt, die bei der History- und Schnell-Ersetzung verwendet werden. Der Default ist !^.
histdup={all|prev}	Bei **all** werden Ereignisse in der History nur je einmal aufgezeichnet, bei **prev** wird ein neues Ereignis nicht aufgezeichnet, wenn es mit dem vorigen identisch ist.
histfile=Datei	Gibt die History-Datei an, die von **history -S** und **history -L** verwendet wird. Der Default ist *~/.history*.
histlit	Wenn diese Variable gesetzt ist, werden Zeilen aus der History bei ihrem Aufruf nicht expandiert.
history=*n Format*	Das erste Wort gibt an, wie viele Befehle in der History-Liste abzuspeichern sind. Das zweite Wort gibt das Format an, in dem die History-Liste angezeigt wird (nur in der **tcsh**, im Abschnitt über **Prompts** werden die möglichen Formate erläutert).
home=*Dir*	Das Heimatverzeichnis des Benutzers. Der Initialwert wird aus der Variablen HOME genommen. Diese Variable kann auch durch ~ abgekürzt werden.
ignoreeof	Ignoriert ein Dateiendezeichen (*EOF*) an Terminals und verhindert so unabsichtliches Abmelden.

inputmode= {insert|overwrite}	Einstellen des Editiermodus.
listjobs=long	Wenn ein Job angehalten wird, werden alle Jobs angezeigt (in langem Format, falls der Wert long ist).
listlinks	Beim **ls-F**-Befehl wird auch der Typ der Datei angezeigt, auf die ein Link zeigt.
listmax=Num	**list-choices** gibt nicht mehr als *Num* Auswahlmöglichkeiten vor der Eingabeaufforderung aus.
listmaxrows=Num	**list-choices** gibt nicht mehr als *Num* Reihen mit Auswahlmöglichkeiten vor der Eingabeaufforderung aus.
loginsh	Ist gesetzt, falls es sich um eine Anmeldeshell handelt.
logout	Gibt den Zustand der bevorstehenden Abmeldung an (**normal**, **automatic** oder **hangup**).
mail=(*n Dateien*)	Enthält die Dateien, die alle fünf Sekunden (oder alle *n* Sekunden, falls *n* gesetzt ist) auf neue Mail überprüft werden.
matchbeep= {never|nomatch| ambiguous|notunique}	Gibt an, wann im Falle einer Vervollständigung ein akustisches Signal ausgegeben wird: nie (**never**), wenn keine Vervollständigung möglich ist (**nomatch**), wenn mehrere möglich sind (**ambiguous**) oder wenn mehrere möglich sind und eine davon exakt ist (**notunique**).
nobeep	Schaltet akustische Signale ab.
noclobber	Ausgabeumleitungen können nicht auf bereits existierende Dateien zeigen. So kann ein unabsichtliches Überschreiben solcher Dateien verhindert werden.
noglob	Abschalten der Dateinamenexpansion; nützlich in Shell-Skripten.
nokanji	Abschalten der Kanji-Unterstützung (wenn vorhanden).
nonomatch	Metazeichen in Dateinamen werden als literale Zeichen betrachtet, wenn kein Dateiname paßt. So wird beispielsweise bei **vi ch*** die neue Datei **ch*** angelegt, anstatt "No match" auszugeben.
nostat=Verzeichnisliste	Während der Vervollständigung wird *Verzeichnisliste* nicht gelesen.
notify	Mitteilung ausgeben, wenn ein Job beendet wurde.
owd	Letztes Arbeitsverzeichnis.
path=(*Dirs*)	Liste der Pfadnamen von Verzeichnissen, in denen ausführbare Dateien gesucht werden. Der Wert wird aus der Variablen PATH genommen, der Default ist `. /usr/ucb /usr/bin`.
printexitvalue	Ausgeben von Rückgabewerten, die von Null verschieden sind.
prompt='*Str*'	String, der als Eingabeaufforderung für interaktive Eingaben verwendet wird. Der Default ist %. Siehe den Abschnitt „Formatieren der prompt-Variablen" weiter unten.
prompt2='Str'	String, der als Eingabeaufforderung für interaktive Eingaben in **foreach**- und **while**-Schleifen sowie Fortsetzungszeilen (d.h. Zeilen nach gequoteten Zeilenwechseln) verwendet wird. Siehe den Abschnitt „Formatieren der prompt-Variablen" weiter unten.

prompt3='Str'	String, der als Eingabeaufforderung für interaktive Eingaben während der Rechtschreibprüfung verwendet wird. Siehe den Abschnitt „Formatieren der prompt-Variablen" weiter unten.
pushdtohome	Wechsel zum Heimatverzeichnis, wenn **pushd** ohne Argumente aufgerufen wird.
pushdsilent	Verzeichnisstack bei Aufruf von **pushd** und **popd** nicht ausgeben.
recexact	Vervollständigung gilt beim ersten passenden Namen als abgeschlossen.
recognize_only_ executables	Bei der Vervollständigung von Befehlen nur ausführbare Dateien ausgeben.
rmstar	Sicherheitsabfrage vor der Ausführung von **rm** *.
savedirs	**dirs -S** vor dem Verlassen der Shell ausführen.
savehist=*Max* [merge]	Vor dem Verlassen der Shell **history -S** ausführen und nicht mehr als *Max* Zeilen abspeichern. Wenn **merge** angegeben wird, werden die Zeilen mit denen früherer Speicherungen zusammengefaßt und nach Uhrzeit sortiert.
sched=string	Formatierung der Ereignis-Ausgabe von **sched**. Siehe Abschnitt „Formatieren der prompt-Variablen" weiter unten.
shell=*Datei*	Pfadname der aktuellen Shell; der Default ist */bin/tcsh*.
shlvl	Anzahl verschachtelter Shells.
showdots[=-A]	Anzeigen versteckter Dateien bei **ls -F**. Wenn **-A** angegeben wird, werden **.** und **..** nicht angezeigt.
status=*n*	Rückgabewert des letzten Befehls. Eingebaute Befehle geben 0 (erfolgreich) oder 1 (nicht erfolgreich) zurück.
symlinks= {chase|ignore|expand}	Gibt an, wie symbolische Links behandelt werden sollen: Bei *cwd* zu den tatsächlichen Verzeichnisnamen expandieren (**chase**), als existierende Verzeichnisse behandeln (**ignore**) oder Argumente, die wie Pfadnamen aussehen, expandieren (**expand**).
tcsh	Version der **tcsh**.
term	Terminaltyp.
time=*'n %c'*	Wenn die Ausführung eines Befehls mehr als *n* CPU-Sekunden benötigt, werden Benutzerzeit, Systemzeit, verstrichene Zeit und CPU-Ausnutzung ausgegeben. Mit *%c* können weitere Daten ausgegeben werden.
tperiod	Anzahl von Minuten, nach denen wieder der **periodic**-Alias ausgeführt wird.
tty	Name des tty, wenn vorhanden.
uid	Benutzer-ID.
user	Benutzername.
verbose	Nach einer History-Ersetzung wird der neue Befehl angezeigt; wie **csh -v**.
version	Versionsnummer und zusätzliche Informationen über die Shell (u.a. die beim Kompilieren eingestellten Optionen).

visiblebell	Aufblitzen des Bildschirms anstelle eines akustischen Signals.
watch=([n] Benutzer Terminal...)	Überprüfen, ob der *Benutzer* am *Terminal* angemeldet ist, wobei *Terminal* der Name eines ttys oder **any** sein kann. Die Überprüfung findet alle *n* Minuten statt, der Default ist 10.
who=String	Angeben der Information, die bei **watch** ausgegeben wird.
wordchars=Zeichenliste	Liste von nicht-alphanumerischen Zeichen, die als Teil eines Wortes akzeptiert werden. Der Default ist `*?_-.[]~=`.

Formatieren der prompt-Variablen

tcsh bietet eine Reihe von Ersetzungen an, die in der Eingabeaufforderung verwendet werden können (in der **csh** stehen normaler Text und History-Substitution zur Verfügung).

%%	Wörtliches %
%/	Aktuelles Arbeitsverzeichnis
%~	Aktuelles Arbeitsverzeichnis in ~-Schreibweise
%#	# beim Superuser, > bei allen anderen
%?	Rückgabewert des letzten Befehls
%b	Fettdruck aufheben
%c[[0]*n*], %.[[0]*n*]	Die letzten *n* (Default 1) Teile des aktuellen Arbeitsverzeichnisses. Wenn 0 angegeben wird, werden die weggelassenen Teile durch `/<skipped>` ersetzt.
%d	Wochentag (z.B. Mon, Tue)
%h, %!, !	Nummer des aktuellen Ereignisses in der History
%l	Aktuelles tty
%m	Erster Teil des Rechnernamens
%n	Benutzername
%p	Aktuelle Zeit mit Sekunden in zwölfstündiger Darstellung
%s	Inverse Darstellung aufheben
%t, %@	Aktuelle Zeit in zwölfstündiger Darstellung
%u	Unterstreichung aufheben
%w	Monat (z.B. Jan, Feb)
%y	Jahr (z.B. 94, 95)
%B	Fettdruck einschalten
%C	Wie %c, aber mit vollen Pfadnamen anstelle der ~-Schreibweise
%D	Tag im Monat (z.B. 09, 10)
%M	Vollqualifizierter Rechnername
%P	Aktuelle Zeit mit Sekunden in 24stündiger Darstellung
%S	Einschalten der inversen Darstellung
%T	Aktuelle Zeit in 24stündiger Darstellung
%U	Einschalten der Unterstreichung
%W	Monat (z.B. 09, 10)
%Y	Jahr (z.B. 1994, 1995)

Beispielhafte .cshrc-Datei

```
# VORDEFINIERTE Variablen

set path=(~ ~/bin /usr/ucb /bin /usr/bin . )
set mail=(/usr/mail/tom)

if ($?prompt) then             # Einstellungen fuer interaktives Arbeiten
  set echo
  set noclobber ignoreeof

  set cdpath=(/usr/lib /usr/spool/uucp)
# Jetzt kann cd macros anstelle von
# cd /usr/lib/macros eingegeben werden

  set history=100
  set prompt='kalle \!% '       # History-Nummer mit ausgeben
  set time=3

# MEINE VARIABLEN

  set man1="/usr/man/man1"     # ermoeglicht  cd $man1, ls $man1
  set a="[a-z]*"               # ermoeglicht  vi $a
  set A="[A-Z]*"               # oder         grep string $A

# ALIASE

  alias c "clear; dirs"        # Anfuehrungsstriche schuetzen ; und |
  alias h "history|more"
  alias j jobs -l
  alias ls ls -sFC             # ls-Befehl umdefinieren
  alias del 'mv \!* ~/tmp_dir' # sicherer als rm
endif
```

Umgebungsvariablen

Die C-Shell verwaltet eine Reihe von *Umgebungsvariablen*, die sich von den Shell-Variablen unterscheiden und eigentlich kein Teil der C-Shell sind. Shell-Variablen haben ihre Bedeutung nur in der aktuellen Shell, während Umgebungsvariablen automatisch exportiert werden und somit global verfügbar sind. So sind beispielsweise C-Shell-Variablen nur in dem Skript ansprechbar, in dem sie definiert worden sind, wohingegen Umgebungsvariablen auch von anderen Shell-Skripten, Mail-Programmen oder Editoren verwendet werden können.

So werden Umgebungsvariablen Werte zugewiesen:

`setenv` *`VAR Wert`*

Per Konvention werden Umgebungsvariablen in Großbuchstaben geschrieben. Sie können eigene Umgebungsvariablen anlegen oder die vordefinierten, weiter unten beschriebenen Umgebungsvariablen verwenden.

Die folgenden Umgebungsvariablen korrespondieren mit C-Shell-Variablen. Wenn sich der Wert in einer von beiden ändert, wird er auch in die andere kopiert:

HOME	Heimatverzeichnis; wie **home**.
PATH	Suchpfad für Befehle; wie **path**.
TERM	Terminaltyp; wie **term**.

Andere vordefinierte Umgebungsvariablen sind unter anderem (Namen in kursiver Schrift bezeichnen **tcsh**-spezifische Variablen):

EXINIT Ein String von **ex**-Befehlen ähnlich denen in der *.exrc*-Datei (z.B. **set ai**). Wird von **vi** und **ex** verwendet.

HOST Name des Rechners.

LOGNAME Ein anderer Name für die USER-Variable.

MAIL Der Name der Datei, in der die Mail abgespeichert wird. Wird von Mail-Programmen verwendet. Diese Datei ist nicht identisch mit der **mail**-Shell-Variablen, die nur auf neue Mail prüft.

OSTYPE Betriebssystem.

PWD Das aktuelle Verzeichnis, der Wert wird aus **cwd** kopiert.

REMOTEHOST Rechnername des entfernten Rechners.

SHELL Defaultmäßig undefiniert.

TERMCAP Der Name der Datei, welche die Befehle zum Positionieren des Cursors für das von Ihnen verwendete Terminal enthält. Der Default ist */etc/termcap*.

Ausdrücke

Ausdrücke werden bei @, **if**- und **while**-Anweisungen verwendet, um z.B. arithmetische Berechnungen, Stringvergleiche oder Dateitests durchzuführen. Auch bei **exit** und **set** können Ausdrücke angegeben werden. Ausdrücke werden durch Kombination von Variablen und Konstanten mit Operatoren angegeben, die denen der Programmiersprache C ähneln. Die Präzedenz der Operatoren ist gleich der in C. Die Faustregel ist:

1. `* / %`
2. `+ -`

Alle anderen Ausdrücke sollten mit ( ) gruppiert werden. Klammern müssen verwendet werden, wenn der Ausdruck <, >, & oder | enthält.

Operatoren

Operatoren sind von einem der folgenden Typen:

Zuweisungsoperatoren

`=`	Wertzuweisung
`+=  -=`	Addition/Subtraktion und anschließende Zuweisung
`*=  /=  %=`	Multiplikation/Division/Restbildung und anschließende Zuweisung
`&=  ^=  |=`	Bitweises UND/XOR/OR und anschließende Zuweisung
`++`	Inkrementierung
`--`	Dekrementierung

Arithmetische Operatoren

`*  /  %`	Multiplikation, Ganzzahl-Division, Restbildung
`+  -`	Addition, Subtraktion

Bitweise und logische Operatoren

`~` Binäres Inverse (Einerkomplement)

`!` Logische Negation

`<<   >>` Bitweises Verschieben nach links/rechts

`&` Bitweises UND

`^` Bitweises exklusives ODER

`|` Bitweises ODER

`&&` Logisches UND

`||` Logisches ODER

`{ Befehl }` Gibt 1 zurück, wenn *`Befehl`* erfolgreich war, sonst 0. Dies ist genau andersherum als der normale Rückgabewert von *`Befehl`*. Meistens ist es praktischer, die **$status**-Variable zu verwenden.

Vergleichsoperatoren

`==   !=` Gleichheit, Ungleichheit

`<=   >=` Kleiner gleich, größer gleich

`<   >` Kleiner, größer

Dateitestoperatoren

Bevor der Dateitest ausgeführt wird, werden Befehlsersetzung und Dateinamenexpansion auf *`Datei`* ausgeführt.

`-d` *`Datei`*	*`Datei`* ist ein Verzeichnis.
`-e` *`Datei`*	*`Datei`* existiert.
`-f` *`Datei`*	*`Datei`* ist eine normale Datei.
`-o` *`Datei`*	*`Datei`* gehört dem Benutzer.
`-r` *`Datei`*	Der Benutzer darf *`Datei`* lesen.
`-w` *`Datei`*	Der Benutzer darf *`Datei`* beschreiben.
`-x` *`Datei`*	Der Benutzer darf *`Datei`* ausführen.
`-z` *`Datei`*	*`Datei`* hat die Länge 0.
`!`	Dreht die Bedeutung der obigen Operatoren um.

Beispiele

Die folgenden Beispiele zeigen @-Befehle und setzen **`n`** = 4 voraus.

```
Ausdruck                           Wert von $x
@ x = ($n > 10 || $n < 5)              1
@ x = ($n >= 0 && $n < 3)              0
@ x = ($n << 2)                       16
@ x = ($n >> 2)                        1
@ x = $n % 2                           0
@ x = $n % 3                           1
```

Die folgenden Beispiele geben jeweils die erste Zeile einer **if**- oder **while**-Anweisung wieder:

Ausdruck	Bedeutung
`while ($#argv != 0)`	Solange noch Argumente vorhanden sind . . .
`if ($today[1] == "Frei")`	Wenn das erste Wort „Frei“ ist . . .
`if (-f $argv[1])`	Wenn das erste Argument eine normale Datei ist . . .
`if (! -d $tmpdir)`	Wenn `tmpdir` kein Verzeichnis ist . . .

Befehls-History

Früher ausgeführte Befehle werden in einer sogenannten History-Liste ausgegeben. Sie können auf diese Liste zugreifen, um Befehle zu überprüfen, erneut auszuführen oder zu verändern. Der eingebaute **history**-Befehl zeigt die History-Liste an. Außerdem beeinflussen die vordefinierten Variablen **histchars** und **history** den History-Mechanismus. Die History-Liste kann auf dreierlei Art und Weise benutzt werden:

- Befehls-Ersetzungen (mit ! und ^).
- Argument-Ersetzungen (bestimmte Wörter in einem Befehl).
- Verwenden von Modifikatoren, um einzelne Teile eines Befehls oder Wortes zu extrahieren oder zu ersetzen.

Befehls-Ersetzung

`!`	Einleiten einer Befehls-Ersetzung
`!!`	Der letzte Befehl
`!`*`N`*	Befehl Nr. *N* in der History-Liste
`!-`*`N`*	Der *N*-te Befehl vor dem aktuellen
`!`*`String`*	Der letzte mit *`String`* anfangende Befehl
`!?`*`String`*`?`	Der letzte *`String`* enthaltende Befehl
`!?`*`String`*`?%`	Das letzte Befehlsargument, das *`String`* enthält
`!$`	Das letzte Argument des vorhergehenden Befehls
`!!`*`String`*	Der letzte Befehl, an den *`String`* angehängt wird
`!`*`N String`*	Befehl Nr. *N*, an den *`String`* angehängt wird
`!{`*`s1`*`}`*`s2`*	Der letzte mit *`s1`* beginnende Befehl, an den *`s2`* angehängt wird
`^`*`alt`*`^`*`neu`*`^`	Schnell-Ersetzung: Im letzten Befehl wird *`alt`* durch *`neu`* ersetzt und der so entstandene Befehl ausgeführt

Beispiele für Befehls-Ersetzung

Angenommen wird jeweils der folgende Befehl:

```
%3 vi cprogs/01.c kap002 kap03
```

Befehls-nummer	Eingegebener Befehl	Ausgeführter Befehl
4	`^00^0`	`vi cprogs/01.c kap02 kap03`
5	`nroff !*`	`nroff cprogs/01.c kap02 kap03`
6	`nroff !$`	`nroff kap03`
7	`!vi`	`vi cprogs/01.c kap02 kap03`
8	`!6`	`nroff kap03`
9	`!?01`	`vi cprogs/01.c kap02 kap03`
10	`!{nr}.new`	`nroff kap03.new`
11	`!!|lp`	`nroff kap03.new | lp`
12	`more !?pr?%`	`more cprogs/01.c`

Wort-Ersetzung

Jeder Angabe darf ein Doppelpunkt vorangestellt werden.

`:0`	Name des Befehls
`:`*n*	Argument Nr. *n*
`^`	Erstes Argument
`$`	Letztes Argument
`:`*n*`-`*m*	Argumente *n* bis *m*
`-`*m*	Wörter 0 bis *m*; wie :0-*m*
`:`*n*`-`	*n*-tes bis vorletztes Argument
`:`*n*`*`	*n*-tes bis letztes Argument; wie *n*-$
`*`	Alle Argumente; wie ^-$ or 1-$
`#`	Aktuelle Kommandozeile bis zu dieser Stelle; ziemlich nutzlos

Beispiele für Wort-Ersetzungen

Angenommen wird jeweils der folgende Befehl:

```
%13 cat kap01 kap02 kap03 biblio rueckseite
```

Befehls-nummer	Eingegebener Befehl	Ausgeführter Befehl
14	`ls !13^`	`ls kap01`
15	`sort !13:*`	`sort kap01 kap02 kap03 biblio rueckseite`
16	`lp !cat:3*`	`more kap03 biblio rueckseite`
17	`!cat:0-3`	`cat kap01 kap02 kap03`
18	`vi !-5:4`	`vi biblio`

History-Modifikatoren

Befehls- und Wort-Ersetzungen können mit einer oder mehreren der folgenden Anweisungen modifiziert werden:

Anzeigen, Ersetzen und Quoten

`:p`	Befehl anzeigen, aber nicht ausführen.
`:s/`*`alt`*`/`*`neu`*	Erstes Auftreten von String *`alt`* durch *`neu`* ersetzen.
`:gs/`*`alt`*`/`*`neu`*	Jedes Auftreten von *`alt`* durch *`neu`* ersetzen.
`:&`	Erstes Auftreten der letzten Ersetzung (`:s` oder ^-Befehl) wiederholen.
`:g&`	Jedes Auftreten der letzten Ersetzung wiederholen.
`:q`	Wortliste quoten.
`:x`	Einzelne Wörter quoten.

Löschen

`:r`	Dateinamenextension entfernen.
`:gr`	Alle Dateinamenextensionen entfernen.
`:e`	Alles außer der ersten Extension entfernen.
`:ge`	Alle Extensionen entfernen.
`:h`	Alles nach dem letzten Schrägstrich entfernen. Zurück bleibt der Pfadname ohne Dateiname.
`:gh`	Alle Dateinamen entfernen.
`:t`	Alles bis zum letzten Schrägstrich entfernen. Zurück bleibt der Dateiname ohne Pfad.
`:gt`	Alle Pfadnamen entfernen und nur die Dateinamen zurücklassen.
`:u`	Ersten kleinen Buchstaben groß schreiben (nur **tcsh**).
`:l`	Ersten großen Buchstaben klein schreiben (nur **tcsh**).
`:a`	Den folgenden Modifikator so oft wie möglich auf ein Wort anwenden. Wenn `a` zusammen mit `g` verwendet wird, wird `a` auf jedes Wort angewendet (nur **tcsh**).

csh und tcsh

Beispiele für History-Modifikatoren

Befehl Nr. 17 sei:

```
%17 cat kap01 kap02 kap03
```

Befehls-nummer	Eingegebener Befehl	Ausgeführter Befehl
19	`!17:s/kap/KAP/`	`cat KAP01 kap02 kap03`
20	`!17g&`	`cat KAP01 KAP02 KAP03`
21	`!more:p`	`more cprogs/01.c` *(nur angezeigt)*
22	`cd !$:h`	`cd cprogs`
23	`vi !mo:$:t`	`vi 01.c`
24	`grep stdio !$`	`grep stdio 01.c`
25	`^stdio^include stdio^:q`	`grep "include stdio" 01.c`
26	`nroff !21:t:p`	`nroff 01.c` *(Wollte ich das wirklich?)*
27	`!!`	`nroff 01.c` *(ausführen)*

Manipulation der Kommandozeile

Vervollständigung

Sowohl **tcsh** als auch **csh** bieten die Vervollständigung von Wörtern. Die **tcsh** vervollständigt Wörter und Befehle automatisch, wenn auf die **TAB**-Taste gedrückt wird; die **csh** vervollständigt Wörter und Befehle, wenn die Shell-Variable *filec* gesetzt ist und auf **ESC** gedrückt wird. Wenn die Vervollständigung mehrdeutig ist (d.h. auf die eingegebenen Zeichen mehr als eine Vervollständigung paßt), vervollständigt die Shell soviel wie möglich und gibt ein akustisches Signal aus, um anzuzeigen, daß die Vervollständigung nicht abgeschlossen werden konnte. Eine Liste aller möglichen Vervollständigungen kann durch Drücken von **CTRL-D** abgerufen werden. In der **tcsh** wird eine abgeschlossene Vervollständigung durch Anhängen eines Leerzeichens an vollständige Dateinamen oder Befehle und eines Schrägstriches an vollständige Verzeichnisse angezeigt.

Sowohl **csh** als auch **tcsh** erkennen die ~-Schreibweise zur Bezeichnung des Heimatverzeichnisses. Die Shells nehmen an, daß Wörter am Anfang der Kommandozeile und solche, die unmittelbar auf `|`, `&`, `;`, `||` und `&&` folgen, Befehle sind, und passen den Suchpfad entsprechend an. Vervollständigung kann auch in der Mitte eines Wortes aufgerufen werden; es werden nur die Zeichen links vom Cursor herangezogen. In diesem Fall gibt **CTRL-D** die Liste der möglichen Vervollständigungen nur bei **vi**-Tastenbelegung, nicht aber bei Emacs-Tastenbelegung aus.

Verwandte Shell-Variablen

- autolist
- fignore
- listmax
- listmaxrows

Verwandte Befehle zum Editieren der Kommandozeile

- complete-word-back
- complete-word-forward
- expand-glob
- list-glob

Verwandte eingebaute Shell-Befehle

- complete
- uncomplete

Editieren der Kommandozeile

In der **tcsh** können Sie sich mit dem Cursor in der Kommandozeile bewegen und Befehle editieren. Es gibt zwei Modi zum Editieren der Kommandozeile, die auf den am meisten verwendeten Texteditoren basieren, Emacs-Modus und **vi**-Modus.

Zwischen den beiden Modi kann folgendermaßen umgeschaltet werden:

`bindkey -e`	*Emacs-Tastaturbelegung*
`bindkey -v`	*vi-Tastaturbelegung*

Der wesentliche Unterschied zwischen Emacs- und **vi**-Tastaturbelegungen besteht darin, daß Emacs-Tastaturbelegungen keine Modi kennen, d.h., die einzelnen Befehle stehen immer zur Verfügung. Im **vi**-Modus muß dagegen zwischen Einfüge- und Befehls-Modi hin und her geschaltet werden; in jedem Modus stehen andere Befehle zur Verfügung. Weitere Unterschiede:

- Der Emacs-Modus ist einfacher; mit dem **vi**-Modus können feinere Einstellungen vorgenommen werden.
- Mit dem Emacs-Modus können Texte ausgeschnitten und wieder eingefügt werden; außerdem können Markierungen gesetzt werden. Der **vi**-Modus verfügt nicht über diese Funktionen.
- Die Fähigkeiten zum Durchsuchen der Befehls-History unterscheiden sich.

Emacs-Modus

Die folgenden Tabellen beschreiben die verschiedenen Tastaturbefehle, die im Emacs-Modus zur Verfügung stehen.

Befehle zum Positionieren des Cursors (Emacs-Modus)

Befehl	Beschreibung
CTRL-B	Cursor ein Zeichen nach links bewegen.
CTRL-F	Cursor ein Zeichen nach rechts bewegen.
ESC b	Cursor ein Wort nach links bewegen.
ESC f	Cursor ein Wort nach rechts bewegen.
CTRL-A	Cursor an den Zeilenanfang bewegen.
CTRL-E	Cursor an das Zeilenende bewegen.

Befehle zum Löschen von Text (Emacs-Modus)

Befehl	Beschreibung
DEL oder CTRL-H	Löschen des Zeichens links vom Cursor.
CTRL-D	Löschen des Zeichens unter dem Cursor.
ESC d	Löschen des Wortes unter dem Cursor.
ESC DEL oder ESC CTRL-H	Löschen des Wortes links vom Cursor.
CTRL-K	Von der Cursorposition bis zum Zeilenende löschen.
CTRL-U	Ganze Zeile löschen.

Bearbeitung von Befehlen (Emacs-Modus)

Befehl	Beschreibung
CTRL-P	Vorheriger Befehl.
CTRL-N	Nächster Befehl.
Pfeil nach oben	Vorheriger Befehl.
Pfeil nach unten	Nächster Befehl.
Befehlsfragment ESC p	History nach *Befehlsfragment* durchsuchen. *Befehlsfragment* muß dabei am Anfang eines Befehls stehen.
Befehlsfragment ESC n	Wie **ESC p**, aber mit vorwärts gerichteter Suche.
esc *Num*	Nächsten Befehl *Num*-mal wiederholen.
CTRL-Y	Zuletzt gelöschten String einfügen.

vi-Modus

Im **vi**-Modus gibt es zwei Submodi: den Einfüge- und den Befehlsmodus. Der Einfügemodus ist der Default. Zwischen den beiden Modi kann mit ESC hin und her geschaltet werden; außerdem kann durch Eingabe von **a** (append) oder **i** (insert) vom Befehlsmodus zum Eingabemodus geschaltet werden.

Befehle, die im Einfüge- und Befehlsmodus zur Verfügung stehen (vi-Modus)

Befehl	Beschreibung
CTRL-P	Vorheriger Befehl.
CTRL-N	Nächster Befehl.
Pfeil nach oben	Vorheriger Befehl.
Pfeil nach unten	Nächster Befehl.
ESC	Umschalten zwischen den Modi.

Befehle zum Editieren (vi-Einfügemodus)

Befehl	Beschreibung
CTRL-B	Cursor ein Zeichen nach links bewegen.
CTRL-F	Cursor ein Zeichen nach rechts bewegen.
CTRL-A	Cursor an den Zeilenanfang bewegen.
CTRL-E	Cursor an das Zeilenende bewegen.
DEL oder CTRL-H	Zeichen links vom Cursor löschen.
CTRL-W	Wort links vom Cursor löschen.
CTRL-U	Vom Zeilenanfang bis zur Cursorposition löschen.
CTRL-K	Von der Cursorposition bis zum Zeilenende löschen.

Befehle zur Cursorpositionierung (vi-Befehlsmodus)

Befehl	Beschreibung
h oder **CTRL-H**	Cursor ein Zeichen nach links bewegen.
l oder **Leertaste**	Cursor ein Zeichen nach rechts bewegen.
w	Cursor ein Wort nach links bewegen.
b	Cursor ein Wort nach rechts bewegen.
e	Cursor an das nächste Wortende bewegen.
W, B, E	Wie **w**, **b** und **e**, behandelt aber nur Leerzeichen, Tabulatorzeichen und Zeilenwechsel als Worttrenner anstelle von allen nicht-alphanumerischen Zeichen.
^ or **CTRL-A**	Cursor an den Zeilenanfang bewegen (erstes Zeichen, das weder Leerzeichen noch Tabulator noch Zeilenwechsel ist).
0	Cursor an den Zeilenanfang bewegen.
$ or **CTRL-E**	Cursor an das Zeilenende bewegen.

Befehle zum Einfügen von Text (vi-Befehlsmodus)

Befehl	Beschreibung
a	So lange Text hinter dem Cursor einfügen, bis auf **ESC** gedrückt wird.
i	So lange Text vor dem Cursor einfügen, bis auf **ESC** gedrückt wird.
A	So lange Text nach dem Zeilenende einfügen, bis auf **ESC** gedrückt wird.
I	So lange Text vor dem Zeilenanfang einfügen, bis auf **ESC** gedrückt wird.

Befehle zum Löschen von Text (vi-Befehlsmodus)

Befehl	Beschreibung
x	Zeichen unter dem Cursor löschen.
X or **DEL**	Zeichen links vom Cursor löschen.
d*m*	Bis zum Ende des Positionierungsbefehls *m* löschen.
D	Wie **d$**.
CTRL-W	Ein Wort zurück löschen.
CTRL-U	Vom Zeilenanfang bis zur Cursorposition löschen.
CTRL-K	Von der Cursorposition bis zum Zeilenende löschen.

Befehle zum Ersetzen von Text (vi-Befehlsmodus)

Befehl	Beschreibung
c*m*	Alle Zeichen von der Cursorposition bis zum Ende des Positionierungsbefehls *m* ersetzen, bis auf ESC gedrückt wird.
C	Wie **c$**.
r*c*	Zeichen unter dem Cursor durch *c* ersetzen.
R	Mehrere Zeichen ersetzen, bis auf **ESC** gedrückt wird.
s	Zeichen unter dem Cursor durch eingegebene Zeichen ersetzen, bis auf **ESC** gedrückt wird.

Nach Zeichen suchende Positionierungsbefehle (vi-Befehlsmodus)

Befehl	Beschreibung
f*c*	Cursor zum nächsten Auftreten von *c* in der Zeile bewegen.
F*c*	Cursor zum vorherigen Auftreten von *c* in der Zeile bewegen.
t*c*	Cursor vor das nächste Auftreten von *c* in der Zeile bewegen.
T*c*	Cursor hinter das vorherige Auftreten von *c* in der Zeile bewegen.
;	Letzten f- oder F-Befehl wiederholen.
,	Letzten f- oder F-Befehl in umgekehrter Richtung wiederholen.

Eingebaute csh- und tcsh-Befehle

@	@ [*Variable*[*n*]=*Ausdruck*] Zuweisen des Wertes des arithmetischen Ausdrucks an *Variable* oder das *n*-te Element von *Variable*, wenn der Index *n* angegeben wird. Wenn weder *Variable* noch *Ausdruck* angegeben werden, werden die Werte aller Shell-Variablen ausgegeben (wie **set**). Die möglichen Operatoren für Ausdrücke sind unter „Ausdrücke" weiter vorn in diesem Kapitel aufgeführt. Außerdem existieren zwei besondere Formen: @ *Variable*++ Wert von *Variable* um 1 erhöhen. @ *Variable*–– Wert von *Variable* um 1 vermindern.
#	# Ignorieren des gesamten Textes, der auf derselben Zeile folgt. # wird in Shell-Skripten als Kommentarzeichen verwendet und ist eigentlich kein Befehl.
#!	#!*Shell* Wird als erste Zeile eines Shell-Skriptes verwendet und gibt an, daß das Skript von der *Shell* (mit optionalen Argumenten) ausgeführt werden soll. Wird nicht von allen Shells unterstützt. Beispiel: `#!/bin/csh -f`
:	: Leerer Befehl; der Rückgabewert ist 0. Der Doppelpunkt-Befehl wird oft als erstes Zeichen eines Bourne- oder Korn-Shell-Skriptes verwendet, um zu verhindern, daß ein # (Doppelkreuz) unbeabsichtigt das erste Zeichen ist.
alias	**alias** [*Name* [*Befehl*]] Legt *Name* als Abkürzung oder Alias für *Befehl* fest. Wenn *Befehl* nicht angegeben wird, wird die aktuelle Auflösung von *Name* ausgegeben, wenn auch *Name* weggelassen wird, werden alle Abkürzungen ausgegeben.

Abkürzungen können auf der Kommandozeile eingegeben werden, sie werden jedoch meistens in der Datei *.cshrc* verwendet, damit sie direkt nach dem Einloggen zur Verfügung stehen (siehe auch die beispielhafte *.cshrc*-Datei weiter vorn in diesem Kapitel). Abkürzungsdefinitionen können ähnlich wie die History-Liste auch Kommandozeilenargumente ansprechen. Mit \!* werden alle Kommandozeilenargumente angesprochen, mit \!^ das erste, mit \!\!:2 das zweite, mit \!$ das letzte usw. Die Abkürzung *Name* darf auch ein UNIX-Befehl sein, allerdings geht dann die ursprüngliche Bedeutung des Befehls verloren, es sei denn, es wird *Name* eingegeben. Siehe auch **unalias** und den Abschnitt „Besondere Abkürzungen in der tcsh".

alias

Beispiele

Setzen der Größe von **xterm**-Fenstern im X Window System:

```
alias R 'set noglob; eval `resize`; unset noglob'
```

Anzeigen von Abkürzungen, die den String **ls** enthalten:

```
alias | grep ls
```

Ausführen von **nroff** auf allen Kommandozeilenargumenten:

```
alias ms 'nroff -ms \!*'
```

Kopieren der Datei, die als erstes Argument angegeben wird:

```
alias back 'cp \!^ \!^.old'
```

Verwenden des originalen Befehls **ls** anstelle der Abkürzung:

```
% \ls
```


alloc

alloc

Ausgeben der Summen des benutzten und freien Speichers.

bg [*JobIDs*]

bg

Bringt den aktuellen oder die durch *JobIDs* angegebenen Jobs in den Hintergrund.

Beispiel

Um einen lang andauernden Prozeß in den Hintergrund zu setzen, könnte zunächst folgender Befehl eingegeben werden:

```
4% nroff -ms report CTRL-Z
```

und danach einer der folgenden:

```
5% bg
5% bg %         Aktueller Job
5% bg %1        Job Nr. 1
5% bg %nr       Paßt auf den Zeilenanfang nroff
5% % &
```

bindkey	**bindkey** [*Optionen*] [*Taste*] [*Befehl*] Nur in der **tcsh**. Zeigt alle Tastenbindungen an oder bindet eine Taste an einen Befehl. -a Ausgeben der Standard- und alternativen Tastenbindungen. -b *Taste* Erwartet, daß *Taste* eines der folgenden Dinge ist: ein Control-Zeichen (in Zircumflex-Notation, z.B. ^B, oder in C-Notation, z.B. C-B), ein Metazeichen (z.B. M-B), eine Funktionstaste (z.B. F-*String*) oder eine erweiterte Präfix-Taste (z.B. X-B). -c *Befehl* Interpretiert *Befehl* als Shell-Befehl (und nicht als Editor-Befehl). -d *Taste* Bindet die Taste an ihre originale Bindung. -e Stellt die Standard-Emacs-Bindungen ein. -k *Taste* Erwartet, daß *Taste* eine Cursortaste ist (**left**, **right**, **up** oder **down**). -l Ausgeben aller Editor-Befehle mit Beschreibungen. -r *Taste* Alle Bindungen für *Taste* löschen. -v Stellt die Standard-vi-Bindungen ein.
break	**break** Setzt die Ausführung mit dem Befehl fort, der auf den **end**-Befehl folgt, der die innerste **while**- oder **foreach**-Schleife beendet.
breaksw	**breaksw** Beendet eine **switch**-Anweisung, die Ausführung wird nach der **endsw**-Anweisung fortgesetzt.
built-ins	**built-ins** Nur in der **tcsh**. Ausgeben aller eingebauten Shell-Befehle.
bye	**bye** Nur in der **tcsh**. Wie **logout**.
case	**case** *Suchmuster*: Gibt ein *Suchmuster* in der **switch**-Anweisung an.
cd	**cd** [*Dir*] Wechselt das aktuelle Verzeichnis nach *Dir*. Der Default ist das Heimatverzeichnis des Benutzers. Wenn *Dir* ein relativer Pfadname ist, der aber nicht

im aktuellen Verzeichnis ist, wird die Variable **cdpath** durchsucht. Siehe auch die beispielhafte **.cshrc**-Datei weiter vorn in diesem Kapitel. In der **tcsh** gibt es noch die folgenden Optionen:

-	Wechselt in das vorhergehende Verzeichnis.
-l	~-Notation wird ausdrücklich expandiert.
-p	Verzeichnisstack ausgeben.

cd

chdir [*dir*]

chdir

Wie **cd**. Nützlich, wenn Sie **cd** umdefiniert haben.

complete [*String* [*Wort*/*Suchmuster*/*Liste*[**:** *Auswahl*]/[*Suffix*]]]

complete

Nur in der **tcsh**. Gibt alle Vervollständigungen oder, wenn *String* angegeben ist, alle Vervollständigungen für *String* aus. *String* darf ein Suchmuster sein. Mit *Wort* können noch weitere Optionen angegeben werden:

c	Nur das aktuelle Wort vervollständigen, *Suchmuster* ignorieren.
C	Nur das aktuelle Wort vervollständigen, dabei *Suchmuster* verwenden.
n	Das vorherige Wort vervollständigen.
N	Das Wort vor dem vorherigen vervollständigen.
P	Erwartet, daß *Suchmuster* ein Zahlenbereich ist. Vervollständigung wird in diesem Bereich durchgeführt.

Es können mehrere *Listen* nach Vervollständigungen durchsucht werden. Einige der *Listen*-Optionen:

(*string*)
Elemente der Liste *String*

\`*Befehl*\`
Ausgabe von *Befehl*

a	Abkürzungen
b	Bindungen
c	Befehle
C	Externe (nicht eingebaute) Befehle
d	Verzeichnisse
D	Verzeichnisse, deren Name mit *String* anfängt
e	Umgebungsvariablen
f	Dateinamen
F	Dateinamen, die mit *String* anfangen
g	Gruppen
t	Textdateien
T	Textdateien, deren Name mit *String* anfängt
u	Benutzer-*Auswahl* sollte ein Suchmuster für Dateien sein. Die Vervollständigungen müssen auf dieses Suchmuster passen. An alle Vervollständigungen wird noch *Suffix* angehängt.

continue	**continue** Ausführung der innersten **while**- oder **foreach**-Schleife mit dem nächsten Schleifendurchlauf fortsetzen.
default	**default :** Gibt den Default-Fall (üblicherweise den letzten) in einer **switch**-Anweisung an.
dirs	**dirs** [*Optionen*] Gibt den Verzeichnisstack aus, wobei das aktuelle Verzeichnis als erstes angezeigt wird. Siehe auch **pushd** und **popd**. Alle Optionen mit Ausnahme von -l stehen nur in der **tcsh** zur Verfügung. -l Das Symbol für das Heimatverzeichnis (~) wird zum tatsächlichen Verzeichnisnamen expandiert. -n Ausgabe umbrechen. -v Ein Verzeichnis pro Zeile ausgeben. -L *Datei* Stack aus *Datei* neu aufbauen. *Datei* sollte mit dem Befehl **dirs -S** *Datei* erzeugt worden sein. -S *Datei* Schreibt eine Reihe von **pushd**- und **popd**-Befehlen nach *Datei*. Mit diesen Befehlen kann der Stack später wieder restauriert werden.
echo	**echo** [-n] *String* Ausgeben von *String* auf die Standardausgabe. Wenn **-n** angegeben wird, wird die Ausgabe nicht mit einem Zeilenwechsel abgeschlossen. Im Gegensatz zum **echo**-Befehl in der Bourne-Shell und dem externen Befehl */bin/echo* unterstützt die C-Shell-Version keine Escape-Sequenzen. Siehe auch **echo** in Kapitel 1, *Linux-Benutzerbefehle*, und Kapitel 3, *bash: Die Bourne Again Shell*.
echotc	**echotc** [*Optionen*] *Argumente* Je nach Parametern Anzeigen der Fähigkeiten des Terminals oder Bewegen des Cursors. Es gibt u.a. folgende Argumente: **baud** Anzeige der aktuellen Übertragungsrate in Baud. **cols** Anzeige der aktuellen Zeile. **cm** *Spalte Zeile* Bewegen des Cursors an die angegebene Stelle. **home** Cursor an die linke obere Ecke des Bildschirms bewegen. **lines** Ausgeben der Zeilenanzahl des Bildschirms. **meta** Abfragen, ob das Terminal meta-fähig ist (normalerweise ist die Meta-Taste die ALT-Taste). **tabs** Abfragen, ob das Terminal tabulator-fähig ist.

else

else

Internes Schlüsselwort für die **if** ... **endif**-Anweisung.

end

end

Internes Schlüsselwort, das eine **foreach**-Anweisung beendet.

endif

endif

Internes Schlüsselwort, das eine **if**-Anweisung beendet.

endsw

endsw

Internes Schlüsselwort, das eine **switch**-Anweisung beendet.

eval *Args*

eval

Args ist eine Kommandozeile, die Shell-Variablen enthalten kann. **eval** erzwingt die Expansion der Variablen und führt dann den resultierenden Befehl aus. Diese „doppelte Expansion" ist nützlich, wenn Shell-Variablen Umleitungssymbole, Abkürzungen oder andere Shell-Variablen enthalten. Beispielsweise finden Umleitungen normalerweise vor der Expansion von Variablen statt, so daß eine Variable, die Umleitungssymbole enthält, erst mit **eval** ausgewertet werden muß, weil ansonsten die Umleitungssymbole nicht interpretiert werden. **eval** wird normalerweise nur in Shell-Skripten verwendet.

Beispiele

Die folgende Zeile kann in die **.login**-Datei geschrieben werden, um Terminalcharakteristiken einzustellen:

```
set noglob eval `tset -s xterm` unset noglob
```

Die folgenden Befehle demonstrieren die Funktion von **eval**:

```
% set b='$a'
% set a=Hallo
% echo $b              Einmaliges Lesen der Kommandozeile
$a
% eval echo $b         Zweimaliges Lesen der Kommandozeile
Hallo
```

Ein weiteres Beispiel zur Verwendung von **eval** finden Sie unter **alias**.

exec *Befehl*

exec

Ausführen von *Befehl* anstelle der aktuellen Shell. Damit wird die aktuelle Shell beendet, anstatt daß ein neuer Prozeß für *Befehl* erzeugt wird.

exit

exit [(*Ausdr*)]

Beenden des Shell-Skriptes mit dem Rückgabewert *Ausdr*. Der Rückgabewert 0 bedeutet eine erfolgreiche Ausführung, jeder andere Wert ein Fehlschlagen. Wenn *Ausdr* nicht angegeben wird, wird als Rückgabewert der Wert der **status**-Variablen verwendet. **exit** kann auch auf der Kommandozeile verwendet werden, um das Shell-Fenster zu schließen (also sich ggf. abzumelden).

fg

fg [*JobIDs*]

Holt den aktuellen oder die durch *JobIDs* angegebenen Jobs in den Vordergrund. *JobIDs* kann auch in %*Jobnummer*-Schreibweise angegeben werden.

Beispiel

Wenn Sie eine Editiersitzung mit dem **vi** durch Drücken von **CTRL-Z** unterbrechen, können Sie sie mit einem der folgenden Befehle wieder aufnehmen:

```
% %
% fg
% fg %
% fg %vi          Paßt auf den Anfang des Befehls
```

filetest

filetest *–op Dateien*

Nur in der **tcsh**. Wendet den Dateitestoperator auf *Dateien* an und gibt die Resultate in einer Liste aus. Eine Liste von Dateitestoperatoren finden Sie im entsprechenden Abschnitt weiter vorn in diesem Kapitel.

foreach

foreach *Name* (*Wortliste*)
    Befehle
end

Weist der Variablen *Name* der Reihe nach je einen der Werte in *Wortliste* zu und führt die *Befehle* zwischen **foreach** und **end** aus. **foreach** kann als mehrzeiliger Befehl von der Kommandozeile (erstes der untenstehenden Beispiele) oder in einem Shell-Skript verwendet werden (zweites Beispiel).

Beispiele

Umbenennen aller Dateien, die mit einem Großbuchstaben beginnen:

```
% foreach i ([A-Z]*)
? mv $i $i.neu
? end
```

Bei jedem Kommandozeilenargument überprüfen, ob es sich um eine Option handelt oder nicht:

```
foreach arg ($argv)
   # Faengt das Argument mit - an?
   if ("$arg" =~ -*) then
      echo "Argument ist eine Option"
```

```
    else
       echo "Argument ist ein Dateiname"
    endif
end
```

foreach

glob *Wortliste*

glob

Führt Dateinamen-, Variablen- und History-Ersetzungen auf *Wortliste* durch. Backslash-Sequenzen werden nicht berücksichtigt. Die einzelnen Wörter werden durch Null-Zeichen voneinander getrennt. **glob** wird normalerweise in Shell-Skripten verwendet, um einen Wert dauerhaft vorzugeben, der sich für den Rest des Skriptes nicht mehr ändern soll.

goto *String*

goto

Springt zu der Zeile, deren führende nicht-leere Zeichen *String* gefolgt von einem : sind. Der *String* in der **goto**-Zeile kann auch eine Variable oder ein Dateinamenmuster sein, der Name des Sprungziels muß aber ein konstanter, bereits expandierter Wert sein. Das Sprungziel darf nicht innerhalb einer **foreach**- oder **while**-Schleife stehen.

hashstat

hashstat

Ausgeben einer Statistik, die den Erfolg der Hashtabelle beim Suchen von Befehlen in der **path**-Variable wiedergibt.

history [*Optionen*]

history

Ausgeben der History-Liste. Eine Beschreibung der Syntax finden Sie weiter vorn in diesem Kapitel im Abschnitt „Befehls-History".

Optionen

-c
: Nur in der **tcsh**. Löschen der History-Liste.

-h
: Ausgeben der History-Liste ohne Nummern.

-r
: Ausgeben der Liste in umgekehrter Reihenfolge, die ältesten Befehle erscheinen als letzte.

n
: Nur die letzten *n* Befehle der History ausgeben (Default: der Wert der **history**-Shell-Variablen).

-L *Datei*
: Nur in der **tcsh**. Laden einer Reihe von **pushd**- und **popd**-Befehlen aus *Datei*, mit denen ein abgespeicherter Stack restauriert werden kann.

-M *Datei*
: Nur in der **tcsh**. Zusammenführen des aktuellen Verzeichnisstacks mit dem in *Datei* gespeicherten. Beide werden zeitlich sortiert in *Datei* als Folge von **pushd**- und **popd**-Befehlen abgespeichert.

→

history
←

-S *Datei*

Nur in der **tcsh**. Schreiben einer Reihe von **pushd**- und **popd**-Befehlen nach *Datei*. Aus dieser Datei kann später der aktuelle Verzeichnisstack wieder restauriert werden.

Beispiel

Abspeichern und Ausführen der letzten fünf Befehle:

```
history -h 5 > do_it
source do_it
```

hup

hup [*Befehl*]

Nur in der **tcsh**. Führt *Befehl* aus, der aber abgebrochen wird, wenn er ein Hangup-Signal bekommt. Dieses Signal wird von der Shell bei ihrer eigenen Beendigung gesendet. Wenn *Befehl* nicht angegeben wird, beendet sich das aktuelle Shell-Skript bei einem Hangup-Signal.

if

if

Einleiten einer bedingten Anweisung. Die einfachste Form ist:

```
if (Ausdr) Befehl
```

Es gibt noch drei weitere Befehle:

```
if (Ausdr) then
   Befehle
endif
```

```
if (Ausdr) then
   Befehle1
else
   Befehle2
endif
```

```
if (Ausdr) then
   Befehle1
else if (Ausdr) then
   Befehle2
else
   Befehle
endif
```

In der einfachsten Form wird *Befehl* ausgeführt, wenn *Ausdr* wahr ist, ansonsten passiert nichts (Umleitungen finden trotzdem statt; dies ist ein Fehler der Shell). In den anderen Formen werden jeweils einer oder mehrere Befehle ausgeführt. Wenn *Ausdr* wahr ist, werden die Befehle nach **then** ausgeführt, ansonsten wird entweder zu den Befehlen nach **else** verzweigt, oder es wird hinter **else if** verzweigt, wo eine weitere Abfrage stattfindet. Weitere Beispiele finden Sie im Abschnitt „Ausdrücke" weiter vorn in diesem Kapitel sowie unter **shift** und **while**.

Beispiel

Wenn keine Kommandozeilenargumente angegeben werden, wird ein Default angenommen:

```
if ($#argv == 0) then
   echo -n "Kein Dateiname angegeben. "
   echo "Ausgabe wird nach Report geschrieben."
   set outfile = Report
else
   set outfile = $argv[1]
endif
```

jobs [-l]

jobs

Ausgeben aller laufenden oder angehaltenen Jobs. Mit -l werden auch die Prozeß-IDs ausgegeben. Sie können so beispielsweise überprüfen, ob ein langer Übersetzungs- oder Formatierungsvorgang noch läuft. Auch vor dem Abmelden ist **jobs** nützlich.

kill [*Optionen*] *ID*

kill

Beendet alle Prozesse oder Jobs, die durch *ID* angegeben werden. Der Benutzer muß der Besitzer des Prozesses oder der Superuser sein. Dieser eingebaute Befehl ist ähnlich dem in Kapitel 1 beschriebenen */bin/kill*, erlaubt aber auch die Angabe symbolischer Dateinamen. Prozesse, die sich normal nicht beenden lassen, können mit Hilfe des Signals 9 beendet werden.

Optionen

-l	Ausgeben der Signalnamen (keine *ID* angeben).
-*signal*	Das zu schickende Signal, entweder als Nummer oder als Name. Der Default ist TERM.

Signale

Die auf dem System bekannten Signale stehen in */usr/include/sys/signal.h* und werden hier auszugsweise ohne das SIG-Präfix wiedergegeben:

HUP	1	Verbindung beendet
INT	2	Interrupt
QUIT	3	Beenden
ILL	4	Illegale Maschineninstruktion
TRAP	5	Trace trap
IOT	6	IOT-Instruktion
EMT	7	EMT-Instruktion
FPE	8	Fließkommafehler
KILL	9	Abbruch (kann nicht abgefangen werden)
BUS	10	Adreßfehler (Bus error)
SEGV	11	Segmentfehler (Segmentation violation)
SYS	12	Falsches Argument im Systemaufruf
PIPE	13	Ausgabe in eine Datei, aus der kein Prozeß liest
ALRM	14	Auslösen eines Timers
TERM	15	Beenden (Default-Signal)
USR1	16	Benutzerdefiniertes Signal 1
USR2	17	Benutzerdefiniertes Signal 2
CLD	18	Kind-Prozeß ist gestorben
PWR	19	Neustart nach Stromausfall

Beispiele

Wenn Sie den folgenden Befehl eingegeben haben:

```
44% nroff -ms report &
```

können Sie ihn mit einem der folgenden Befehle beenden:

→

kill
←

```
45% kill 19536        Prozeß-ID
45% kill %            Aktueller Job
45% kill %1           Job Nr. 1
45% kill %nr          Anfang der Kommandozeile
45% kill %?report     Passender Ausschnitt der Kommandozeile
```

limit

limit [-h] [*Ressource* [*Limit*]]

Ausgeben oder Setzen von Ressourcen-Begrenzungen, die für den aktuellen Prozeß und alle von ihm gestarteten Prozesse gelten. Wenn *Limit* nicht angegeben wird, wird das aktuell für *Ressource* geltende Limit ausgegeben. Wenn *Ressource* ebenfalls weggelassen wird, werden alle Limits ausgegeben. Defaultmäßig werden die aktuellen Limits angezeigt oder gesetzt; wenn -h angegeben wird, werden harte Limits verwendet. Harte Limits stellen eine absolute Grenze dar, die nicht überschritten werden kann. Nur der Superuser kann harte Limits heraufsetzen. Siehe auch **unlimit**.

Option

-h Verwenden von harten Limits.

Ressource

cputime Maximale Anzahl von Sekunden, die ein Prozeß an CPU-Zeit verbrauchen darf. Kann mit **cpu** abgekürzt werden.

filesize Maximale Größe einer Datei.

datasize Maximale Größe des Datenbereichs eines Prozesses (einschließlich Stack).

stacksize Maximale Größe des Stacks.

coredumpsize Maximale Größe einer Core Dump-Datei.

Limit

Limit ist eine Zahl gefolgt von einem optionalen Zeichen, das die Einheit angibt.

Bei **cputime**: *n*h (für *n* Stunden)
*n*m (für *n* Minuten)
mm:ss (Minuten und Sekunden)
Bei allen anderen: *n*k (für *n* Kilobyte, der Default)
*n*m (für *n* Megabyte)

log

log

Nur in der **tcsh**. Abfragen der **watch**-Variablen, welche Benutzer überwacht werden sollen, und Ausgaben einer Liste derjenigen Benutzer davon, die gerade angemeldet sind. Wenn - als Option verwendet wird, wird die Umgebung zurückgesetzt, als hätte sich der Benutzer mit einer neuen Gruppe angemeldet.

login [*Benutzer* | -p] — login

Ersetzen der Anmeldeshell von *Benutzer* durch */bin/login*. Mit **-p** bleiben die Umgebungsvariablen erhalten.

logout — logout

Beenden der Anmeldeshell.

ls-F [*Optionen*] [*Dateien*] — ls-F

Nur in der **tcsh**. Schnellere Alternative zu **ls -F**. Wenn Optionen angegeben werden, wird statt dessen **ls** aufgerufen.

newgrp [–] [*Gruppe*] — newgrp

Nur in der **tcsh**. Ändern der Gruppen-ID des Benutzers in die angegebene Gruppen-ID oder die originale Gruppen-ID, falls keine angegeben ist. Wenn - als Option angegeben wird, wird die Umgebung zurückgesetzt, als ob der Benutzer sich mit einer neuen Gruppe angemeldet hätte.

nice [±*n*] *Befehl* — nice

Ändert die Priorität der Ausführung von *Befehl* oder der aktuellen Shell, wenn *Befehl* nicht angegeben wird. (Siehe auch **nice** in Kapitel 1.) Der Wertebereich für *n* ist –20 bis 20, der Default ist 4. –20 ist dabei die höchste Priorität (schnellste Ausführung), 20 die niedrigste. Nur der Superuser darf einen negativen Wert angeben.

+*n*	*n* zum aktuellen Prioritätswert addieren (Priorität herabsetzen).
-*n*	*n* vom aktuellen Prioritätswert subtrahieren (Priorität heraufsetzen). Kann nur vom Superuser ausgeführt werden.

nohup [*Befehl*] — nohup

Befehl wird ausgeführt und soll nicht beendet werden, wenn die Terminalverbindung endet (d.h., wenn Sie sich abmelden oder die Telefonleitung abbrechen). Wenn *Befehl* nicht angegeben wird, wird das aktuelle Skript nicht beendet. (Siehe auch **nohup** in Kapitel 1.)

notify [*JobID*] — notify

Sofortiges Ausgeben einer Meldung, wenn ein Hintergrund-Job beendet wird (normalerweise wird erst bei der Ausgabe eines Shell-Prompts eine entsprechende Meldung ausgegeben). Wenn keine *JobID* angegeben wird, wirkt der Befehl auf den aktuellen Hintergrund-Job.

onintr

onintr *Marke*
onintr -
onintr

Wird in Shell-Skripten verwendet, um Interrupt-Signale abzufangen (ähnlich dem **trap 2** bzw. **trap "" 2**-Befehl in der **bash**). Die erste Form entspricht **goto** *Marke*, die Shell verzweigt nach *Marke:*, wenn sie ein Interrupt-Signal (z.B. durch Drücken von CTRL-C) empfängt. Mit der zweiten Form werden alle Interrupts ignoriert. Dies ist sinnvoll am Anfang eines Shell-Skripts oder eines Abschnitts, der ohne Abbruch laufen muß (z.B. beim Verschieben von Dateien). Mit der dritten Form wird die vor **onintr -** gültige Interrupt-Hantierung wiederhergestellt.

Beispiel

```
onintr cleanup        # Bei einem Interrupt nach "cleanup"
 .                    # springen
 .                    # Befehle im Shell-Skript
 .
cleanup:              # Sprungziel für Interrupts
  onintr -            # Keine Interrupts mehr zulassen
  rm -f $tmpdateien   # Angelegte Dateien löschen
  exit 2              # Skript mit Fehlermeldung beenden
```

popd

popd [*Optionen*]

Entfernen des obersten oder *n*-ten Eintrags vom Verzeichnisstack. Wenn kein Eintrag angegeben wurde, wird in das nach dem Entfernen oben liegende Verzeichnis gesprungen. Danach wird der aktuelle Verzeichnisstack ausgegeben. Der oberste Eintrag hat die Nummer 0 und erscheint bei Ausgaben ganz links. Siehe auch **dirs** oder **pushd**.

Optionen

- +*n* Angeben des *n*-ten Eintrags.
- -l Expandieren der ~-Schreibweise.
- -n Umbrechen langer Zeilen.
- -v Ausgeben genau eines Verzeichnisses pro Zeile.

printenv

printenv [*Variable*]

Ausgeben aller oder der angegebenen Umgebungsvariablen mit ihren Werten.

pushd

pushd *Name*
pushd [*Optionen*]
pushd

In der ersten Form wird in das angegebene Verzeichnis verzweigt, das auch auf den Stack gelegt wird. In der zweiten Form wird der *n*-te Eintrag an die Spitze rotiert und zum aktuellen Verzeichnis gemacht. (Die Numerierung beginnt bei 0.) In der dritten Form vertauscht **pushd** die obersten beiden

Einträge des Verzeichnisstacks und wechselt zum neuen oben liegenden Verzeichnis. Die Optionen +n, -l, -n und -v haben die gleiche Bedeutung wie bei **popd**. Siehe auch **dirs** und **popd**. pushd

Beispiele

`% dirs` `/home/kalle /usr`	
`% pushd /etc` `/etc /home/kalle /usr`	`/etc` *auf den Verzeichnisstack legen*
`% pushd +2` `/usr /etc /home/kalle`	*In das dritte Verzeichnis verzweigen*
`% pushd` `/etc /usr /home/kalle`	*Die oberen beiden Verzeichnisse vertauschen*
`% popd` `/usr /home/kalle`	*Obersten Eintrag verwerfen und zum nächsten verzweigen*

rehash

rehash

Neuberechnen der Hashtabelle für die PATH-Variable. **rehash** muß immer dann verwendet werden, wenn während der aktuellen Sitzung ein neuer Befehl im Pfad erzeugt wird. Erst danach kann mit Hilfe der PATH-Variablen ein Befehl lokalisiert und ausgeführt werden. (Wenn der neue Befehl in einem Verzeichnis steht, das nicht in PATH aufgeführt ist, fügen Sie dieses Verzeichnis zunächst der PATH-Variablen hinzu, bevor Sie **rehash** aufrufen.) Siehe auch **unhash**.

repeat *n Befehl*

repeat

Befehl n-mal ausführen.

Beispiele

memo dreimal ausdrucken:

```
% repeat 3 pr memo | lp
```

Zehn Zeilen vom Terminal einlesen und in **item_list** ablegen:

```
% repeat 10 line > item_list
```

50 Muster-Dateien an **report** anhängen:

```
% repeat 50 cat template >> report
```

sched [*Optionen*]
sched *Zeit Befehl*

sched

Nur in der **tcsh**. Wenn keine Optionen angegeben werden, werden alle zur Ausführung vorgemerkten Befehle ausgegeben. Mit der zweiten Form kann ein Befehl zur Ausführung vorgemerkt werden.

Zeit sollte im Format *hh:mm* angegeben werden (z.B. 13:00).

→

sched ←

Optionen

+*hh:mm*	Der Befehl soll nach *hh:mm* ab jetzt ausgeführt werden.
-*n*	*n*-tes vorgemerktes Element aus der Vormerkungsliste entfernen.

set

set *Variable* = *Wert*
set [*Optionen*] *Variable*[*n*] = *Wert*
set

Weist *Variable* den *Wert* oder, wenn mehrere Werte angegeben werden, die sich aus den Werten ergebende Wortliste zu. Wenn ein Index *n* angegeben wird, wird das *n*-te Wort in *Variable* auf *Wert* gesetzt. (Die Variable muß in diesem Fall zu diesem Zeitpunkt bereits so viele Wörter enthalten.) Wenn keine Argumente angegeben werden, werden Namen und Werte aller gesetzten Variablen ausgegeben. Siehe auch „Vordefinierte Shell-Variablen" weiter vorn in diesem Kapitel.

Optionen

-r	Nur in der **tcsh**. Ausgeben der nicht-schreibbaren Variablen oder die angegebene Variable nicht-schreibbar machen.

Beispiele

```
% set liste=(ja nein vielleicht) Zuweisung einer Wortliste
% set liste[3]=vielleicht   Zuweisung eines Elements an eine Wortliste
% set quote="Gallia est omnis divisa"  Variablenzuweisung
% set x=5 y=10 history=100 Zuweisung mehrerer Variablen
% set leer                  Zuweisung eines leeren Wertes an leer
```

setenv

setenv [*Name* [*Wert*]]

Zuweisung von *Wert* an die Umgebungsvariable *Name*. Per Konvention ist *Name* immer in Großbuchstaben geschrieben. Wenn kein *Wert* angegeben wird, wird der leere Wert zugewiesen. Wenn überhaupt keine Argumente angegeben werden, werden die Namen und Werte aller Umgebungsvariablen ausgegeben. Es ist nicht notwendig, **setenv** für die PATH-Variable zu verwenden, weil PATH auch automatisch bei Veränderungen von **path** gesetzt wird.

settc

settc *Merkmal Wert*

Setzen von Terminalmerkmalen.

setty

setty [*Optionen*] [+|–*Modus*]

Nur in der **tcsh**. Verhindert, daß die Shell die tty-Modi verändert. Per Default gilt dies für den Ausführungs-Modus.

Optionen

+*Modus*
: Ohne Argumente werden alle Modi ausgegeben, die in der angegebenen Menge eingeschaltet sind. Ansonsten wird der angegebene Modus eingeschaltet.

-*Modus*
: Ohne Argumente werden alle Modi ausgegeben, die in der angegebenen Menge ausgeschaltet sind. Ansonsten wird der angegebene Modus ausgeschaltet.

-a
: Ausgeben aller Modi der angegebenen Menge.

-d
: Bezeichnet die Menge des Editier-Modus (wird bei Editierbefehlen verwendet).

-q
: Bezeichnet die Menge des Quote-Modus (wird verwendet, wenn Zeichen wortwörtlich eingegeben werden).

-x
: Bezeichnet die Menge des Ausführungs-Modus (Default, wird verwendet wenn Programme ausgeführt werden).

shift [*Variable*]

Wenn *Variable* angegeben wird, werden die Wörter in der Wortliste der Variablen um eine Stelle weitergeschoben, d.h., *Name*[2] wird *Name*[1]. Wenn kein Argument angegeben wird, werden die Positionsparameter (Kommandozeilenargumente) um eine Stelle weitergeschoben, d.h., **$2** wird **$1** usw. **shift** wird typischerweise in einer **while**-Schleife verwendet. Siehe auch das zusätzliche Beispiel bei **while**.

Beispiel

```
while ($#argv)        # solange noch Argumente da sind
    if (-f $argv[1])
       wc -l $argv[1]
    else
       echo "$argv[1] ist keine normale Datei"
    endif
    shift             # nächstes Argument holen
end
```


source [-h] *Skript*

Lesen und Ausführen der Befehle eines C-Shell-Skripts. Wenn **-h** angegeben wird, werden die Befehle der History-Liste hinzugefügt, aber nicht ausgeführt.

Beispiel

```
source ~/.cshrc
```


stop [*JobIDs*]

Hält den aktuellen oder die durch *JobIDs* angegebenen Hintergrundjob(s) an. Dieser Befehl ist das Gegenstück zu CTRL-Z oder **suspend**.

suspend

suspend

Anhalten des aktuellen Vordergrund-Jobs (wie CTRL-Z). Wird oft verwendet, um einen **su**-Befehl anzuhalten.

switch

switch

Ausführen von Befehlen abhängig vom Wert einer Variablen. **switch** ist einer **if-then-else**-Anweisung vorzuziehen, wenn mehr als drei Auswahlmöglichkeiten zu berücksichtigen sind. Wenn die Variable *String* auf *Suchmuster1* paßt, wird die erste Folge von Befehlen ausgeführt, wenn *String* auf *Suchmuster2* paßt, die zweite Folge usw. Wenn kein Suchmuster paßt, werden die bei **default** angegebenen Befehle ausgeführt. Bei der Angabe von *String* kann Befehls-Ersetzung, Variablen-Ersetzung und Dateinamenexpansion verwendet werden. Suchmuster können mit den Platzhaltersymbolen *, ? und [ ] angegeben werden. Mit **breaksw** wird eine **switch**-Anweisung beendet. Wenn **breaksw** nicht verwendet wird (ein seltener Fall), wird die **switch**-Anweisung fortgesetzt und eine weitere passende Befehlsfolge ausgeführt, bis eine **breaksw**- oder **endsw**-Anweisung erreicht wird. Im folgenden finden Sie die allgemeine Syntax von **switch** zusammen mit einem Beispiel, in dem das erste Kommandozeilenargument verarbeitet wird.

```
switch (string)           switch ($argv[1])
  case Suchmuster1:        case -[nN]:
      Befehle                nroff $Datei | lp
      breaksw                breaksw
  case Suchmuster2:        case -[Pp]:
      Befehle                pr $Datei | lp
      breaksw                breaksw
  case Suchmuster3:        case -[Mm]:
      Befehle                more $Datei
      breaksw                breaksw
      .                    case -[Ss]:
      .                      sort $Datei
      .                      breaksw
  default:                 default:
      Befehle                echo "Fehler -- Option nicht vorhanden"
                             exit 1
      breaksw                breaksw
endsw                     endsw
```

telltc

telltc

Ausgeben aller Terminalmerkmale und ihrer Werte.

time

time [*Befehl*]

Ausführen eines Befehls und Anzeigen, wie lange die Ausführung gedauert hat. Ohne Argumente kann **time** in einem Shell-Skript verwendet werden, um die Ausführungsdauer des Skriptes zu messen.

umask [*nnn*] | umask

Anzeigen der beim Anlegen von Dateien verwendeten Rechtemaske oder Setzen dieser Maske auf den Oktalwert *nnn*. Diese Maske bestimmt, welche Zugriffsbits gesetzt werden. **umask** ist auch ein Standard-Befehl. Beispiele finden Sie beim Eintrag zu **umask** in Kapitel 1.

unalias *Name* | unalias

Entfernt *Name* von der Liste der Abkürzungen. Siehe die Beschreibung zu **alias**.

uncomplete *Suchmuster* | uncomplete

Nur in der **tcsh**. Entfernt mit **complete** angegebene Vervollständigungen.

unhash | unhash

Entfernt die interne Hashtabelle. Die Shell verwendet die Hashtabelle nicht mehr weiter und durchsucht die **path**-Variable nach Befehlen, was länger dauert. Siehe auch **rehash**.

unlimit [*Ressource*] | unlimit

Hebt die Ressourcen-Beschränkung auf *Ressource* auf. Wenn *Ressource* nicht angegeben wird, werden alle Beschränkungen aufgehoben. Wird **-h** angegeben, werden die harten Limits aufgehoben. Dies kann nur vom Superuser ausgeführt werden.

unset *Variablen* | unset

Entfernt eine oder mehrere *Variablen*. Die Variablen können als Suchmuster unter Verwendung von Platzhalterzeichen für Dateinamen angegeben werden. Siehe auch **set**.

unsetenv *Variable* | unsetenv

Entfernen einer Umgebungsvariablen. Es dürfen *keine* Suchmuster angegeben werden. Siehe auch **setenv**.

wait | wait

Hält die Programmausführung an, bis alle Kindprozesse beendet sind oder der Prozeß ein Interrupt-Signal erhält.

watchlog | watchlog

Wie **log**.

where	where *Befehl* Nur in der **tcsh**. Ausgeben aller Abkürzungen, eingebauten und sonstigen Befehle, die *Befehl* heißen.
which	which *Befehl* Nur in der **tcsh**. Gibt an, welche Version eines Befehls ausgeführt wird. Entspricht dem Linux-Befehl **which**, ist aber schneller und berücksichtigt auch eingebaute **tcsh**-Befehle.

while	while (*Ausdruck*)
	    Befehle
	end
	
	Solange *Ausdruck* wahr ist, werden die *Befehle* zwischen **while** und **end** ausgeführt. Mit **break** und **continue** kann eine Schleife beendet oder neu gestartet werden. Siehe auch das Beispiel bei **shift**.
	
	Beispiel
	
	`set benutzer = (elke linus kalle michael reinhard)`
	`while ($argv[1] != $benutzer[1])`
	`   #Durchlaufen aller Benutzer, überprüfen, ob einer passt`
	`   shift benutzer`
	`   #Wenn alle Benutzer ohne Treffer durchlaufen wurden...`
	`   if ($#benutzer == 0) then`
	`     echo "$argv[1] steht nicht in der Benutzerliste"`
	`     exit 1`
	`   endif`
	`end`

Besondere Abkürzungen (Aliase) in der tcsh

In der **tcsh** können bestimmte Aliasnamen mit Sonderbedeutungen gesetzt werden. Die zugehörigen Befehle werden unter bestimmten Voraussetzungen ausgeführt.

beepcmd Wenn ein akustisches Signal ausgegeben wird.

cwdcmd Wenn **cwd** sich ändert.

periodic Alle paar Minuten. Der genaue Abstand wird durch die *tperiod*-Variable angegeben.

precmd Bevor eine neue Eingabeaufforderung ausgegeben wird.

shell *Shell* Wenn ein Skript keine Shell zur Ausführung angibt, wird es mit *Shell* ausgeführt.

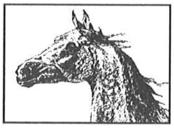
Job-Verwaltung

Mit Hilfe der Befehle zur Job-Verwaltung können Vordergrundjobs in den Hintergrund gebracht, Hintergrundjobs in den Vordergrund geholt und laufende Jobs unterbrochen (vorübergehend angehalten) werden. Die C-Shell stellt die hier angegebenen Befehle zur Job-Verwaltung bereit. Weitere Informationen zu diesen Befehlen finden Sie in „Eingebaute csh- und tcsh-Befehle“ weiter vorn in diesem Kapitel.

bg	Bringt einen Job in den Hintergrund.
fg	Holt einen Job in den Vordergrund.
jobs	Gibt die aktiven Jobs aus.
kill	Beendet einen Job.
notify	Gibt eine Meldung aus, wenn ein Hintergrundjob beendet wird.
stop	Hält einen Hintergrundjob an.
CTRL-Z	Hält einen Vordergrundjob an.

Viele Befehle zur Job-Verwaltung haben eine *JobID* als Argument. Dieses Argument kann folgendermaßen angegeben werden:

%*n*	Job Nr. *n*.
%*s*	Job, dessen Kommandozeile mit *s* anfängt.
%?*s*	Job, dessen Kommandozeile *s* enthält.
%%	Aktueller Job.
%	Aktueller Job (wie oben).
%+	Aktueller Job (wie oben).
%-	Vorheriger Job.



KAPITEL 5

Mustervergleiche

Eine Reihe von Linux-Werkzeugen zur Textbearbeitung ermöglicht es Ihnen, nicht nur nach festen, vorgegebenen Zeichenketten, sondern auch nach Mustern im Text zu suchen und diese manchmal auch zu ersetzen. Zu diesen Werkzeugen gehören die Programme **ed**, **ex**, **vi** und **sed**, die Skriptsprachen **awk** und **Perl**, die Befehle **grep** und **egrep** sowie die im Editor Emacs eingebaute Programmiersprache Elisp. Suchmuster enthalten neben normalen Zeichen auch spezielle Zeichen, die sogenannten Metazeichen.

In diesem Kapitel werden die folgenden Punkte besprochen:

- Dateinamen und Muster
- Verzeichnis der in den einzelnen Programmen zur Verfügung stehenden Metazeichen
- Beschreibung der Metazeichen
- Beispiele

Eine ausführliche Anleitung finden Sie in *Reguläre Ausdrücke* von Jeffrey Friedl.

Dateinamen und Muster

In Mustervergleichen verwendete Metazeichen unterscheiden sich von denen, die beim Suchen von Dateinamen verwendet werden. Wenn Sie einen Befehl auf der Kommandozeile eingeben, werden die speziellen Zeichen erst von der Shell und danach von dem aufgerufenen Programm gelesen. Nicht gequotete Zeichen werden von der Shell zur Expansion von Dateinamen verwendet. So könnte beispielsweise der Befehl

```
$ grep [A-Z]* chap[12]
```

von der Shell als

```
$ grep Array.c Bug.c Comp.c chap1 chap2
```

interpretiert werden. Damit würde **grep** versuchen, das Suchmuster „Array.c" in den Dateien *Bug.c*, *Comp.c*, *chap1* und *chap2* zu suchen. Damit die Metazeichen nicht von der Shell,

sondern von **grep** ausgewertet werden können, müssen sie gequotet werden:

```
$ grep "[A-Z]*" chap[12]
```

Doppelte Anführungszeichen reichen in den meisten Fällen, mit einfachen sind Sie aber immer auf der sicheren Seite. Beachten Sie auch, daß * und ? eine geringfügig andere Bedeutung bei Mustervergleichen und der Expansion von Dateinamen haben.

Metazeichen in Linux-Programmen

Einige Metazeichen sind nur in manchen Programmen bekannt. Die Metazeichen, die im jeweiligen Programm bekannt sind, sind in der folgenden Tabelle mit einem kleinen Quadrat (■) markiert. Eine vollständige Beschreibung der einzelnen Metazeichen folgt in den nächsten Abschnitten. Beachten Sie bitte, daß in Emacs den meisten Metazeichen selbst wiederum ein Backslash vorangestellt werden muß. Konsultieren Sie dazu ggf. die Emacs-Dokumentation oder das Nutshell-Handbuch *Learning GNU Emacs* von Debra Cameron, Bill Rosenblatt und Eric Raymond. Außerdem verfügt speziell Perl über sehr viel mehr Metazeichen, als hier aufgeführt werden können. Weitere Informationen zu Perl finden Sie in *Programmieren mit Perl* von Larry Wall, Randal L. Schwartz und Tom Christiansen.

Symbol	*ed*	*ex*	*vi*	*sed*	*awk*	*grep*	*egrep*	*Emacs*	*Perl*	*Aktion*
`.`	■	■	■	■	■	■	■	■	■	Paßt auf jedes Zeichen.
`*`	■	■	■	■	■	■	■	■	■	Paßt auf beliebig viele vorher angegebene Zeichen.
`^`	■	■	■	■	■	■	■	■	■	Paßt auf einen Zeilenanfang.
`$`	■	■	■	■	■	■	■	■	■	Paßt auf ein Zeilenende.
`\`	■	■	■	■	■	■	■	■		Hebt die Sonderbedeutung des folgenden Zeichens auf.
`[ ]`	■	■	■	■	■	■	■	■	■	Paßt auf ein Zeichen aus einem Bereich oder einer Menge.
`\( \)`	■	■		■				■	■	Speichert den Text, auf den ein Muster paßt, zur späteren Wiederverwendung.
`\{ \}`	■			■		■				Paßt auf eine bestimmte Anzahl von Instanzen.
`\< \>`		■	■					■		Paßt auf einen Wortanfang oder ein Wortende.
`+`					■		■	■	■	Paßt auf eines oder mehrere der vorher angegebenen Zeichen.
`?`					■		■	■	■	Paßt auf keines oder eines der vorher angegebenen Zeichen.
`|`					■		■	■	■	Trennt Auswahlmöglichkeiten.
`( )`					■		■	■	■	Gruppiert Auswahlmöglichkeiten.

In **ed**, **ex** und **sed** muß sowohl ein Suchmuster (links) als auch ein Ersetzungsmuster (rechts) angegeben werden. In **Emacs** und **Perl** werden Ersetzungsmuster je nach verwendetem Befehl angegeben oder nicht. Die Metazeichen in dieser Tabelle sind nur sinnvoll in Suchmustern.

In **ed**, **ex**, **sed**, **Emacs** und **Perl** stehen die folgenden zusätzlichen Metazeichen zur Verfügung, die nur in Ersetzungsmustern erlaubt sind:

Symbol	*ex*	*sed*	*ed*	*Emacs*	*Perl*	*Aktion*
`\`	■	■	■	■		Hebt die Sonderbedeutung des folgenden Zeichens auf.
`\n`	■	■	■	■	■	Setzt einen vorher mit \(\) abgespeicherten Text ein (das, worauf das Suchmuster paßte).
`&`	■	■				Verwendet das vorhergehende Suchmuster noch einmal.
`~`	■					Verwendet das vorhergehende Ersetzungsmuster noch einmal.
`\u \U`	■				■	Ändert alle Kleinbuchstaben in Großbuchstaben.
`\l \L`	■				■	Ändert alle Großbuchstaben in Kleinbuchstaben.
`\E`	■				■	Hebt ein vorhergehendes \U oder \L wieder auf.
`\e`	■				■	Hebt ein vorhergehendes \u oder \l wieder auf.

Metazeichen

Die folgenden Zeichen haben nur in Suchmustern eine gesonderte Bedeutung:

`.` Paßt auf *genau ein* beliebiges Zeichen, außer auf einen Zeilenwechsel.

`*` Paßt auf beliebig viele Vorkommen (auch gar keines) des unmittelbar davorstehenden Zeichens. Das davorstehende Zeichen kann wiederum ein regulärer Ausdruck sein, so bezeichnet z.B. `.` (Punkt) ein beliebiges Zeichen und `.*` „beliebig viele beliebige Zeichen".

`^` Der folgende reguläre Ausdruck muß am Zeilenanfang passen.

`$` Der vorhergehende reguläre Ausdruck muß am Zeilenende passen.

`[ ]` Paßt auf jedes der in den eckigen Klammern enthaltenen Zeichen.

Ein Bindestrich bezeichnet eine Reihe von aufeinanderfolgenden Zeichen. Wenn ein Zircumflex als erstes Zeichen steht, wird die Bedeutung umgedreht, und der reguläre Ausdruck paßt auf jedes Zeichen, das *nicht* in den eckigen Klammern zu finden ist. Ein Bindestrich oder eine schließende eckige Klammer als erstes Zeichen werden als Listenelement aufgefaßt, alle anderen Zeichen an jeder beliebigen Position ohnehin.

`[^ ]` Paßt nicht auf das oder die enthaltene(n) Zeichen.

`\{n,m\}` Paßt auf eine bestimmte Anzahl von unmittelbar vorhergehenden Zeichen, wobei es sich wiederum um reguläre Ausdrücke handeln darf. Die erlaubte Anzahl ergibt sich aus den Zahlen innerhalb der geschweiften Klammern: \{*n*\} paßt auf genau *n* Vorkommen, \{*n*,\} paßt auf mindestens *n* Vorkommen und \{*n*,*m*\} auf jede Anzahl von Vorkommen zwischen *n* und *m*. *n* und *m* müssen zwischen 0 und 256 einschließlich liegen.

`\` Abschalten der Sonderbedeutung des folgenden Zeichens.

`\( \)` Abspeichern des vom Muster zwischen den Klammern abgedeckten Textes in einer gesonderten Zwischenablage. Pro Zeile können bis zu neun Textteile gespeichert werden. Sie können in Ersetzungsmustern mit den Escape-Sequenzen \1 bis \9 wieder eingesetzt werden.

`\< \>` Paßt auf Wortanfänge (\<) und -enden (\>).

`+` Paßt auf einen oder mehrere der vorhergehenden regulären Ausdrücke.

`?`	Paßt auf keinen oder einen der vorherigen regulären Ausdrücke.
`|`	Paßt auf einen der vor und nach dem Strich angegebenen regulären Ausdrücke.
`( )`	Gruppiert reguläre Ausdrücke.

Die folgenden Zeichen haben nur in Ersetzungsmustern eine Sonderbedeutung:

`\`	Abschalten der Sonderbedeutung für das folgende Zeichen.
`\`*n*	Einsetzen des *n*-ten mit `\(` und `\)` abgespeicherten Textes. *n* ist eine Zahl zwischen 1 und 9, wobei 1 den am weitesten links stehenden Ausdruck bezeichnet.
`&`	Verwendet das Suchmuster als Teil des Ersetzungsmusters.
`~`	Verwendet das vorhergehende Ersetzungsmuster als Teil des jetzigen Ersetzungsmusters.
`\u`	Ändert das erste Zeichen des Ersetzungsmusters in einen Großbuchstaben.
`\U`	Ändert sämtliche Kleinbuchstaben im Ersetzungsmuster in Großbuchstaben.
`\l`	Ändert das erste Zeichen des Ersetzungsmusters in einen Kleinbuchstaben.
`\L`	Ändert sämtliche Großbuchstaben im Ersetzungsmuster in Kleinbuchstaben.

Beispiele zum Suchen

Wenn reguläre Ausdrücke mit **grep**, **egrep** oder **Emacs Lisp** verwendet werden, müssen sie von Anführungszeichen umgeben werden. (Wenn das Suchmuster ein $-Zeichen enthält, müssen Sie einfache Anführungszeichen verwenden, wie z.B. *'Muster'*.) Bei Verwendung mit **ed**, **ex**, **sed**, **awk** und **Perl** werden die regulären Ausdrücke üblicherweise in /-Zeichen eingeschlossen, aber jedes andere Trennzeichen funktioniert auch. Einige Beispiele für Suchmuster:

Suchmuster	*Paßt auf?*
`Tiger`	Die Zeichenkette *Tiger*.
`^Tiger`	*Tiger* am Zeilenanfang.
`Tiger$`	*Tiger* am Zeilenende.
`^Tiger$`	*Tiger* ist das einzige Wort auf der Zeile.
`[tT]iger`	*Tiger* oder *tiger*.
`T[aeiou]ger`	Der zweite Buchstabe ist ein Vokal.
`T[^aeiou]ger`	Der zweite Buchstabe ist ein Konsonant, Großbuchstabe oder Symbol.
`T.ger`	Der zweite Buchstabe ist ein beliebiges Zeichen.
`^...$`	Jede Zeile, die genau drei Zeichen enthält.
`^\.`	Jede Zeile, die mit einem Punkt beginnt.
`^\.[a-z][a-z]`	Desgleichen, gefolgt von zwei Kleinbuchstaben (z.B. troff-Requests).
`^\.[a-z]\{2\}`	Desgleichen, aber nur in **grep** und **sed** möglich.
`^[^.]`	Jede Zeile, die nicht mit einem Punkt beginnt.
`Tiger*`	*Tiger*, *Tigerin*, *Tigerhöhle* etc.
`"Wort"`	Ein Wort in doppelten Anführungsstrichen.
`"*Wort"*`	Ein Wort mit oder ohne doppelte(n) Anführungsstriche(n).
`[A-Z][A-Z]*`	Ein oder mehrere Großbuchstabe(n).
`[A-Z]+`	Desgleichen, aber nur in **egrep** und **awk** möglich.
`[A-Z].*`	Ein Großbuchstabe, gefolgt von beliebig vielen (auch null) Zeichen.
`[A-Z]*`	Beliebig viele (auch null) Großbuchstaben.

Suchmuster	*Paßt auf?*
[a-zA-Z]	Ein beliebiger Buchstabe.
[0-9A-Za-z]*	Jede alphanumerische Zeichenfolge.

egrep- oder awk-Suchmuster	*Paßt auf?*
[567]	Eine der Ziffern *5*, *6* oder *7*.
fuenf|sechs|sieben	Eines der Wörter fünf, sechs oder sieben.
80[23]?86	*8086*, *80286* oder *80386*
Bab(y|ies)	*Baby* oder *Babies*

ex- oder vi-Suchmuster	*Paßt auf?*
\<Ball	Wörter wie *Ballhaus* oder *Ball*
und\>	Wörter wie *Hintergrund* oder *und*
\<Tiger\>	Das Wort *Tiger*

sed- oder grep-Suchmuster	*Paßt auf?*
0\{5,\}	Fünf oder mehr Nullen hintereinander
[0-9]\{3\}-[0-9]\{3\}-[0-9]\{2\}	Bankleitzahl (*nnn-nnn-nn*)

Beispiele zum Suchen und Ersetzen

Die folgenden Beispiele demonstrieren die in **sed** und **ex** und teilweise auch in **Perl** und **Emacs** zur Verfügung stehenden Metazeichen zum Suchen und Ersetzen. Beachten Sie bitte, daß **ex**-Befehle mit einem Doppelpunkt beginnen. Ein Leerzeichen wird durch □, ein Tabulatorzeichen durch *tab* dargestellt.

Befehl	*Ergebnis*
s/.*/(&)/	Ganze Zeile wiedergeben, aber in runde Klammern eingeschlossen.
s/.*/mv & &.alt/	Ändern einer Wortliste (ein Wort pro Zeile) in **mv**-Befehle.
/^$/d	Leerzeilen löschen.
:g/^$/d	Desgleichen, in **ex**.
/^[□*tab*]*$/d	Leerzeilen und solche, die nur Leer- oder Tabulatorzeichen enthalten, löschen.
:g/^[□*tab*]*$/d	Desgleichen, in **ex**.
s/□□*/□/g	Mehrere Leerzeichen zu einem einzigen komprimieren.
:%s/□□*/□/g	Desgleichen, in **ex**.
:s/[0-9]/Position &:/	Ziffer in Positionsbeschriftung (auf der aktuellen Zeile) umwandeln.
:s	Erste Ersetzung wiederholen.
:&	Desgleichen.
:sg	Desgleichen, aber für alle Vorkommen auf einer Zeile.
:&g	Desgleichen.
:%&g	Ersetzung global wiederholen.
:.,$s/Prolog/\U&/g	Wort von der aktuellen bis zur letzten Zeile in Großbuchstaben umwandeln.

Befehl	*Ergebnis*
`:%s/.*/\L&/`	Alle Großbuchstaben in der Datei in Kleinbuchstaben umwandeln.
`:s/\<./\u&/g`	Ersten Buchstaben jedes Wortes auf der aktuellen Zeile in Großbuchstaben umwandeln. (Nützlich für Überschriften.)
`:%s/Ja/Nein/g`	Global ein Wort (`Ja`) durch ein anderes (`Nein`) ersetzen.
`:%s/Ja/~/g`	Global ein anderes Wort durch `Nein` ersetzen (vorherige Ersetzung).

Zum Schluß noch ein paar **sed**-Beispiele, wie man Wörter vertauschen kann. Eine einfache Lösung könnte so aussehen:

`s/essen oder trinken/trinken oder essen/` *Wörter vertauschen*

Viel cleverer ist es aber, einen Zwischenbuffer zu verwenden, wie z.B.:

`s/\([Ee]ssen\) oder \([Tt]rinken\)/\2 oder \1/` *Vertauschen mit Zwischenbuffern*

KAPITEL 6

Der Editor Emacs

In diesem Kapitel werden die folgenden Punkte behandelt:

- Einführung
- Typische Probleme
- Emacs-Befehlsübersicht nach Gruppen sortiert
- Emacs-Befehlsübersicht nach Tastenkombinationen sortiert
- Emacs-Befehlsübersicht nach Befehlsnamen sortiert

Einführung

Emacs gibt es in vielen verschiedenen Versionen; dieses Buch beschreibt GNU-Emacs von der Free Software Foundation. Mehr Informationen zum GNU-Emacs finden Sie in *Learning GNU Emacs*, zweite Auflage, von Debra Cameron, Bill Rosenblatt und Eric Raymond.

Emacs wird aufgerufen mit dem Befehl:

> **emacs** [*Datei(en)*]

Auf manchen Rechnern wird GNU auch durch Eingabe von „gmacs" anstelle von „emacs" aufgerufen.

Typische Probleme

Ein mitunter auftretendes Problem besteht darin, daß die Entf- oder Rückschritt-Taste nicht das Zeichen vor dem Cursor löscht wie geplant. Dies liegt an einem inkompatiblen, weil falsch eingestellten Terminal. Eine in vielen Fällen gut verwendbare Lösung besteht darin, eine Datei namens *.emacs* in Ihrem Home-Verzeichnis anzulegen (oder eine zu bearbeiten, die bereits vorhanden ist) und dort die folgenden Zeilen einzugeben:

```
(keyboard-translate ?\C-h ?\C-?)
(keyboard-translate ?\C-\\ ?\C-h)
```

Jetzt sollte das Löschen mit der Entf- und der Rückschritt-Taste funktionieren. Die Hilfe kann nun mit C-\, einer willkürlich ausgewählten Tastenkombination, aufgerufen werden.

Ein weiteres potentielles Problem besteht darin, daß die Tastenkombination C-s auf manchen Terminals die Datenübertragung zum angeschlossenen Rechner anhält, ein Relikt eines altertümlichen Protokolls zur Synchronisierung zwischen Terminal und Rechner. Die Verbindung kann zwar mit C-q wieder aktiviert werden, aber damit können Sie immer noch keine Befehle eingeben, deren Tastenkombination C-s enthält. Die einzige Lösung (abgesehen natürlich von der Verwendung modernerer Kommunikationsmethoden) liegt darin, neue Tasten-Bindungen anzulegen, die kein C-s enthalten.

Bemerkungen zu den Tabellen

Emacs-Befehle verwenden die Control- und die Meta-Taste. Auf vielen modernen Tastaturen gibt es eine Alt-Taste die als Meta-Taste dient. In diesem Abschnitt bedeutet die Notation C-, daß die Control-Taste zusammen mit der folgenden Taste gedrückt werden muß. Gleiches gilt für M- und die Meta-Taste. Anstelle von Meta können Sie aber auch auf die Escape-Taste drücken, sie wieder loslassen und danach auf die folgende Taste drücken.

In den folgenden Befehlstabellen gibt die erste Spalte die Tastenkombination und die letzte Spalte die ausgeführte Aktion an. Wenn es eine mittlere Spalte gibt, enthält sie den Namen des Befehls. Dieser Befehl kann auch durch Eingabe von M-x, gefolgt vom Befehlsnamen, ausgeführt werden. Wenn Sie unsicher in Bezug auf den Namen sind, können Sie auch auf die Tabulator-, die Return- oder die Leertaste drücken. Emacs zeigt dann alle Befehle an, die mit den bereits eingegebenen Zeichen beginnen.

Weil Emacs so ein mächtiger und funktionsreicher Editor ist, der über Hunderte von Befehlen verfügt, können wir hier nicht alle Befehle aufführen, wenn das Buch den Charakter einer Kurz-Referenz beibehalten soll. Sie können durch die verfügbaren Befehle mit C-h (Hilfe) oder M-x (Befehlsnamen) blättern.

Absolut notwendige Befehle

Wenn Sie gerade anfangen, sich in Emacs einzuarbeiten, finden Sie in dieser kurzen Liste die wichtigsten Befehle:

Bindung	Aktion
C-h	Online Hilfe-System aufrufen.
C-x C-s	Datei (genauer: Buffer) speichern.
C-x C-c	Emacs beenden.
C-x u	Letzten Editier-Befehl rückgängig machen (kann auch wiederholt verwendet werden).
C-g	Aktuelle Operation abbrechen.
C-p	Eine Zeile nach oben bewegen.
C-n	Eine Zeile nach unten bewegen.
C-f	Ein Zeichen nach rechts bewegen.
C-b	Ein Zeichen nach links bewegen.
C-v	Einen Bildschirm nach vorn blättern.
M-v	Einen Bildschirm zurückblättern.
C-s	Inkrementelle Suche.
C-d	Zeichen unter dem Cursor löschen.

Bindung	Aktion
DEL	Zeichen unter dem Cursor löschen.
Rückschritt	Zeichen links vom Cursor löschen.

Befehlsübersicht nach Gruppen sortiert

Zur Erinnerung: Die Tabellen führen die Tastenkombination, den Befehlsnamen und eine Beschreibung der ausgeführten Aktion auf. C- bezeichnet die Control-Taste, M- die Meta-Taste.

Befehle zur Dateibearbeitung

Bindung	Befehlsname	Aktion
C-x C-f	find-file	Datei einlesen.
C-x C-v	find-alternate-file	Eine andere Datei einlesen; ersetzt die mit C-x C-f eingelesene.
C-x i	insert-file	Datei an der Cursor-Position einfügen.
C-x C-s	save-buffer	Datei speichern.
C-x C-w	write-file	Bufferinhalt in eine Datei schreiben.
C-x C-c	save-buffers-kill-emacs	Emacs beenden.
C-z	suspend-emacs	Emacs unterbrechen (kann mit **exit** oder **fg** wieder aktiviert werden).

Befehle zur Cursor-Positionierung

Bindung	Befehl	Aktion
C-f	forward-char	Cursor ein Zeichen vorwärts (nach rechts) bewegen.
C-b	backward-char	Cursor ein Zeichen rückwärts (nach links) bewegen.
C-p	previous-line	Cursor eine Zeile rückwärts (nach oben) bewegen.
C-n	next-line	Cursor eine Zeile vorwärts (nach unten) bewegen.
M-f	forward-word	Cursor ein Wort vorwärts bewegen.
M-b	backward-word	Cursor ein Wort rückwärts bewegen.
C-a	beginning-of-line	Cursor an den Zeilenanfang bewegen.
C-e	end-of-line	Cursor an das Zeilenende bewegen.
M-a	backward-sentence	Cursor einen Satz rückwärts bewegen.
M-e	forward-sentence	Cursor einen Satz vorwärts bewegen.
M-{	backward-paragraph	Cursor einen Absatz rückwärts bewegen.
M-}	forward-paragraph	Cursor einen Absatz vorwärts bewegen.
C-v	scroll-up	Cursor eine Bildschirmseite nach unten (!) bewegen.
M-v	scroll-down	Cursor eine Bildschirmseite nach oben (!) bewegen.
C-x [	backward-page	Cursor eine Seite rückwärts bewegen.
C-x]	forward-page	Cursor eine Seite vorwärts bewegen.
M->	end-of-buffer	Cursor zum Bufferende bewegen.
M-<	beginning-of-buffer	Cursor zum Bufferanfang bewegen.
(keine)	goto-line	Cursor zur Zeile *n* bewegen.
(keine)	goto-char	Cursor zum Zeichen *n* bewegen.
C-l	recenter	Bildschirm so neu zeichnen, daß die aktuelle Zeile in der Bildschirmmitte liegt.
M-*n*	digit-argument	Nächsten Befehl *n*-mal ausführen.
C-u *n*	universal-argument	Nächsten Befehl *n*-mal ausführen.

Befehle zum Löschen

Bindung	Befehl	Aktion
Entf	backward-delete-char	Zeichen links vom Cursor löschen.
C-d	delete-char	Zeichen unter dem Cursor löschen.
M-Entf	backward-kill-word	Vorhergehendes Wort löschen.
M-d	kill-word	Wort unter dem Cursor löschen.
C-k	kill-line	Alles von der Cursor-Position bis zum Zeilenende löschen.
M-k	kill-sentence	Satz unter dem Cursor löschen.
C-x Entf	backward-kill-sentence	Vorhergehenden Satz löschen.
C-y	yank	Gelöschten Text zurückholen.
C-w	kill-region	Markierten Bereich löschen (siehe nächsten Abschnitt)
(keine)	backward-kill-paragraph	Vorhergehenden Absatz löschen.
(keine)	kill-paragraph	Alles von der Cursor-Position bis zum Absatzende löschen.

Absätze und Regionen

Bindung	Befehl	Aktion
C-@	set-mark-command	Anfang oder Ende einer Region markieren.
C-SPACE	(wie davor)	(wie davor)
C-x C-p	mark-page	Seite markieren.
C-x C-x	exchange-point-and-mark	Positionen des Cursors und der Markierung vertauschen.
C-x h	mark-whole-buffer	Buffer markieren.
M-q	fill-paragraph	Absatz neu formatieren.
M-g	fill-region	Region neu formatieren.
M-h	mark-paragraph	Absatz markieren.
M-{	backward-paragraph	Cursor einen Absatz rückwärts bewegen.
M-}	forward-paragraph	Cursor einen Absatz vorwärts bewegen.
(keine)	backward-kill-paragraph	Vorhergehenden Absatz löschen.
(keine)	kill-paragraph	Alles von der Cursor-Position bis zum Absatzende löschen.

Abbrechen und Rückgängigmachen von Befehlen

Bindung	Befehl	Aktion
C-g	keyboard-quit	Aktuellen Befehl abbrechen.
C-x u	advertised-undo	Letzten Editiervorgang rückgängig machen (kann auch wiederholt verwendet werden).
(keine)	revert-buffer	Buffer wieder in den Zustand versetzen, in dem er sich befand, als die Datei das letzte Mal (auch automatisch) gespeichert wurde.

Vertauschungsbefehle

Bindung	Befehl	Aktion
C-t	transpose-chars	Zwei Zeichen vertauschen.
M-t	transpose-words	Zwei Wörter vertauschen.
C-x C-t	transpose-lines	Zwei Zeilen vertauschen.
(keine)	transpose-sentences	Zwei Sätze vertauschen.
(keine)	transpose-paragraphs	Zwei Absätze vertauschen.

Groß-/Kleinschreibungsbefehle

Bindung	Befehl	Aktion
M-c	capitalize-word	Ersten Buchstaben eines Wortes groß schreiben.
M-u	upcase-word	Ganzes Wort in Großbuchstaben schreiben.
M-l	downcase-word	Ganzes Wort in Kleinbuchstaben schreiben.
M- – M-c	negatives Argument; capitalize-word	Ersten Buchstaben des vorhergehenden Wortes groß schreiben.
M- – M-u	negatives Argument; upcase-word	Ganzes vorhergehendes Wort in Großbuchstaben schreiben.
M- – M-l	negatives Argument; downcase-word	Ganzes vorhergehendes Wort in Kleinbuchstaben schreiben.
(keine)	capitalize-region	Alle Initialbuchstaben der Region groß schreiben.
C-x C-u	upcase-region	Ganze Region in Großbuchstaben schreiben.
C-x C-l	downcase-region	Ganze Region in Kleinbuchstaben schreiben.

Befehle zum inkrementellen Suchen

Bindung	Befehl	Aktion
C-s	isearch-forward	Vorwärts gerichtete inkrementelle Suche beginnen oder wiederholen.
C-r	isearch-backward	Rückwärts gerichtete inkrementelle Suche beginnen oder wiederholen.
Return	(keiner)	Erfolgreiche Suche beenden.
C-g	keyboard-quit	Inkrementelle Suche abbrechen; zum Ausgangspunkt zurückkehren.
Entf	(keiner)	Falsches Zeichen des Suchwortes löschen.
M-C-r	isearch-backward-regexp	Rückwärts gerichtete inkrementelle Suche nach regulären Ausdrücken.
M-C-s	isearch-forward-regexp	Vorwärts gerichtete inkrementelle Suche nach regulären Ausdrücken.

Wort-Abkürzungsbefehle

Bindung	Befehl	Aktion
(keine)	abbrev-mode	Wort-Abkürzungsmodus starten oder beenden.
C-x -	inverse-add-global-abbrev	Globale Abkürzung und ihre Definition eingeben.
C-x C-h	inverse-add-local-abbrev	Lokale Abkürzung und ihre Definition eingeben.
(keine)	unexpand-abbrev	Letzte Wort-Abkürzung rückgängig machen.
(keine)	write-abbrev-file	Wort-Abkürzungsdatei abspeichern.
(keine)	edit-abbrevs	Wort-Abkürzungen editieren.
(keine)	list-abbrevs	Wort-Abkürzungen anzeigen.
(keine)	kill-all-abbrevs	Alle Wort-Abkürzungen für die Dauer dieser Sitzung löschen.

Buffermanipulationsbefehle

Bindung	Befehl	Aktion
C-x b	switch-to-buffer	Zum angegebenen Buffer wechseln.
C-x C-b	list-buffers	Bufferliste anzeigen.

Bindung	Befehl	Aktion
C-x k	kill-buffer	Buffer löschen.
(keine)	kill-some-buffers	Für jeden Buffer fragen, ob er gelöscht werden soll.
(keine)	rename-buffer	Buffer umbenennen.
C-x s	save-some-buffers	Für jeden Buffer fragen, ob er gespeichert werden soll.

Fenster-Befehle

Bindung	Befehl	Aktion
C-x 2	split-window-horizontally	Fenster horizontal in zwei aufteilen.
C-x 5	split-window-vertically	Fenster vertikal in zwei aufteilen.
C-x >	scroll-right	Fenster nach rechts scrollen.
C-x <	scroll-left	Fenster nach links scrollen.
C-x o	other-window	Zum anderen Fenster wechseln.
C-x 0	delete-window	Aktuelles Fenster löschen.
C-x 1	delete-other-windows	Alle Fenster außer dem aktuellen löschen.
(keine)	delete-windows-on	Alle zu einem bestimmten Buffer gehörenden Fenster löschen.
C-x ^	enlarge-window	Fenster vergrößern.
(keine)	shrink-window	Fenster verkleinern.
C-x }	enlarge-window-horizontally	Fenster verbreitern.
C-x {	shrink-window-horizontally	Fenster schmaler machen.
M-C-v	scroll-other-window	Anderes Fenster scrollen.
C-x 4 f	find-file-other-window	Datei laden und im anderen Fenster anzeigen.
C-x 4 b	switch-to-buffer-other-window	Buffer im anderen Fenster auswählen.
(keine)	compare-windows	Zwei Buffer vergleichen; ersten Unterschied anzeigen.

Besondere Zeichen im Shell-Modus

Bindung	Befehl	Aktion
C-c C-c	interrupt-shell-subjob	Aktuellen Job abbrechen.
C-c C-d	shell-send-eof	Dateiendezeichen.
C-c C-u	kill-shell-input	Aktuelle Zeile löschen.
C-c C-w	backward-kill-word	Vorhergehendes Wort löschen.
C-c C-z	stop-shell-subjob	Aktuellen Job unterbrechen.

Einrück-Befehle

Bindung	Befehl	Aktion
C-x .	set-fill-prefix	Jeder Zeile des Absatzes vom Zeilenanfang bis zur aktuellen Cursorspalte Zeichen voranstellen. Wird durch Ausführung dieses Befehls mit dem Cursor in Spalte 1 gelöscht.
(keine)	indented-text-mode	Major-Mode. Jeder Tabulator definiert eine neue Einrückung für die folgenden Zeilen.
(keine)	text-mode	Indented-text-mode verlassen und zum normalen Text-Modus zurückkehren.

Bindung	Befehl	Aktion
M-C-\	indent-region	Region so einrücken, daß sie zur ersten Zeile der Region paßt.
M-m	back-to-indentation	Cursor auf das erste Zeichen der Zeile setzen.
M-C-o	split-line	Zeile an der Cursor-Position aufbrechen; bis zur Cursorspalte einrücken.
(keine)	fill-individual-paragraphs	Eingerückte Absätze neu formatieren; Einrückungen beibehalten.

Zentrierungsbefehle

Bindung	Befehl	Aktion
M-s	center-line	Aktuelle Zeile zentrieren.
(keine)	center-paragraph	Aktuellen Absatz zentrieren.
(keine)	center-region	Aktuelle Region zentrieren.

Makrobefehle

Bindung	Befehl	Aktion
C-x (	start-kbd-macro	Makro-Definition anfangen.
C-x)	end-kbd-macro	Makro-Definition beenden.
C-x e	call-last-kbd-macro	Zuletzt definiertes Makro ausführen.
M-*n* C-x e	numerisches Argument call-last-kbd-macro	Zuletzt definiertes Makro *n*-mal ausführen.
C-u C-x (	start-kbd-macro	Zuletzt definiertes Makro ausführen, danach weitere Tasteneingaben hinzufügen.
(keine)	name-last-kbd-macro	Zuletzt angelegtes Makro benennen (vor dem Abspeichern).
(keine)	insert-last-keyboard-macro	Benanntes Makro in eine Datei einfügen.
(keine)	load-file	Gespeicherte Makros laden.
(keine)	*macroname*	Gespeichertes Makro ausführen.
C-x q	kbd-macro-query	Abfrage in Makro-Definition einfügen.
C-u C-x q	(keiner)	Rekursives Editieren in einer Makro-Definition starten.
M-C-c	exit-recursive-edit	Rekursives Editieren beenden.

Detaillierte Hilfefunktionen

Bindung	Befehl	Aktion
C-h a	command-apropos	Welche Befehle gibt es zu diesem Stichwort?
(keine)	apropos	Welche Befehle, Funktionen und Variablen gibt es zu diesem Stichwort?
C-h c	describe-key-briefly	Welcher Befehl wird durch diese Tastenkombination ausgeführt?
C-h b	describe-bindings	Welche Tastenbindungen gibt es für diesen Buffer?
C-h k	describe-key	Welchen Befehl führt diese Tastenkombination aus, und was macht er?
C-h l	view-lossage	Was waren die letzten hundert eingegebenen Zeichen?
C-h w	where-is	Wie lautet die Tastenbindung für diesen Befehl?
C-h f	describe-function	Was macht diese Funktion?
C-h v	describe-variable	Was bedeutet diese Variable, und wie lautet ihr Wert?

Bindung	Befehl	Aktion
C-h m	describe-mode	Ausgabe einer Beschreibung des aktuellen Modus.
C-h s	describe-syntax	Wie lautet die Syntaxtabelle dieses Buffers?

Hilfe-Befehle

Bindung	Befehl	Aktion
C-h t	help-with-tutorial	Starten des Emacs-Lernprogramms.
C-h i	info	Starten des Leseprogramms für Info-Dateien.
C-h n	view-emacs-news	Anzeigen von Neuigkeiten über neue Emacs-Versionen.
C-h C-c	describe-copying	Anzeige der Emacs General Public License.
C-h C-d	describe-distribution	Informationen, wie Emacs von der FSF bestellt werden kann.
C-h C-w	describe-no-warranty	Informationen über die (nicht vorhandene) Garantie von Emacs.

Befehlszusammenfassung nach Tastenkombinationen

Die nächsten beiden alphabetischen Listen nennen eine Reihe von Emacs-Befehlen, jeweils mit Tastaturbindung, Befehlsname und Aktion. C- bezeichnet die Control-, M- die Meta-Taste.

Tastenkombinationen mit Control

Bindung	Befehl	Aktion
C-@	set-mark-command	Anfang (oder Ende) der Region markieren.
C-SPACE	(wie davor)	(wie davor)
C-a	beginning-of-line	Cursor an den Zeilenanfang bewegen.
C-b	backward-char	Cursor ein Zeichen rückwärts (nach links) bewegen.
C-c C-c	interrupt-shell-subjob	Aktuellen Job abbrechen.
C-c C-d	shell-send-eof	Dateiendezeichen.
C-c C-u	kill-shell-input	Aktuelle Zeile löschen.
C-c C-w	backward-kill-word	Vorhergehendes Wort löschen.
C-c C-z	stop-shell-subjob	Aktuellen Job unterbrechen.
C-d	delete-char	Zeichen unter dem Cursor löschen.
C-e	end-of-line	Cursor an das Zeilenende bewegen.
C-f	forward-char	Cursor ein Zeichen vorwärts (nach rechts) bewegen.
C-g	keyboard-quit	Aktuellen Befehl abbrechen.
C-h a	command-apropos	Welche Befehle gibt es zu diesem Begriff?
C-h b	describe-bindings	Welche Tastenbindungen gibt es für diesen Buffer?
C-h C-c	describe-copying	Anzeige der Emacs General Public License.
C-h C-d	describe-distribution	Informationen, wie Emacs von der FSF bestellt werden kann.
C-h C-w	describe-no-warranty	Informationen über die (nicht vorhandene) Garantie von Emacs.
C-h c	describe-key-briefly	Welcher Befehl wird durch diese Tastenkombination ausgelöst?
C-h f	describe-function	Was macht diese Funktion?
C-h i	info	Starten des Leseprogramms für Info-Dateien.
C-h k	describe-key	Welchen Befehl führt diese Tastenkombination aus, und was macht er?
C-h l	view-lossage	Was waren die letzten hundert eingegebenen Zeichen?

Bindung	Befehl	Aktion
C-h m	describe-mode	Ausgabe einer Beschreibung des aktuellen Modus.
C-h n	view-emacs-news	Anzeigen von Neuigkeiten über neue Emacs-Versionen.
C-h s	describe-syntax	Wie lautet die Syntaxtabelle dieses Buffers?
C-h t	help-with-tutorial	Starten des Emacs-Lernprogramms.
C-h v	describe-variable	Was bedeutet diese Variable, und wie lautet ihr Wert?
C-h w	where-is	Wie lautet die Tastenbindung für diesen Befehl?
C-k	kill-line	Alles von der Cursorposition bis zum Zeilenende löschen.
C-l	recenter	Bildschirm so neu zeichnen, daß die aktuelle Zeile in der Bildschirmmitte liegt.
C-n	next-line	Cursor eine Zeile vorwärts (nach unten) bewegen.
C-p	previous-line	Cursor eine Zeile rückwärts (nach oben) bewegen.
C-r	isearch-backward	Rückwärts gerichtete inkrementelle Suche beginnen oder wiederholen.
C-r	(keiner)	Rekursives Editieren anfangen (während einer Suchen-und-Ersetzen Operation).
C-s	isearch-forward	Vorwärts gerichtete inkrementelle Suche beginnen oder wiederholen.
C-t	transpose-chars	Zwei Zeichen vertauschen.
C-u *n*	universal-argument	Nächsten Befehl *n*-mal ausführen.
C-u C-x (	start-kbd-macro	Zuletzt definiertes Makro ausführen, danach weitere Tasteneingaben hinzufügen.
C-u C-x q	(keiner)	Rekursives Editieren in einer Makro-Definition.
C-v	scroll-up	Cursor eine Bildschirmseite nach unten (!) bewegen.
C-w	kill-region	Markierten Bereich löschen (siehe nächster Abschnitt)
C-x (	start-kbd-macro	Makro-Definition anfangen.
C-x)	end-kbd-macro	Makro-Definition beenden.
C-x [	backward-page	Cursor eine Seite rückwärts bewegen.
C-x]	forward-page	Cursor eine Seite vorwärts bewegen.
C-x ^	enlarge-window	Fenster vergrößern.
C-x {	shrink-window-horizontally	Fenster schmaler machen.
C-x }	enlarge-window-horizontally	Fenster verbreitern.
C-x <	scroll-left	Fenster nach links scrollen.
C-x >	scroll-right	Fenster nach rechts scrollen.
C-x -	inverse-add-global-abbrev	Globale Abkürzung und ihre Definition eingeben.
C-x .	set-fill-prefix	Jeder Zeile des Absatzes vom Zeilenanfang bis zur aktuellen Cursorspalte Zeichen voranstellen. Wird durch Ausführung dieses Befehls mit dem Cursor in Spalte 1 gelöscht.
C-x 0	delete-window	Aktuelles Fenster löschen.
C-x 1	delete-other-windows	Alle Fenster außer dem aktuellen löschen.
C-x 2	split-window-horizontally	Fenster horizontal in zwei aufteilen.
C-x 4 b	switch-to-buffer-other-window	Buffer im anderen Fenster auswählen.
C-x 4 f	find-file-other-window	Datei laden und im anderen Fenster anzeigen.
C-x 5	split-window-vertically	Fenster vertikal in zwei aufteilen.
C-x b	switch-to-buffer	Zum angegebenen Buffer wechseln.
C-x C-b	list-buffers	Bufferliste anzeigen.
C-x C-c	save-buffers-kill-emacs	Emacs beenden.
C-x C-f	find-file	Datei einlesen.

Bindung	Befehl	Aktion
C-x C-h	inverse-add-local-abbrev	Lokale Abkürzung und ihre Definition eingeben.
C-x C-l	downcase-region	Ganze Region in Kleinbuchstaben schreiben.
C-x C-p	mark-page	Seite markieren.
C-x C-s	save-buffer	Datei speichern.
C-x C-t	transpose-lines	Zwei Zeilen vertauschen.
C-x C-u	upcase-region	Ganze Region in Großbuchstaben schreiben.
C-x C-v	find-alternate-file	Eine andere Datei einlesen; ersetzt die mit C-x C-f eingelesene.
C-x C-w	write-file	Bufferinhalt in eine Datei schreiben.
C-x C-x	exchange-point-and-mark	Positionen des Cursors und der Markierung vertauschen.
C-x Entf	backward-kill-sentence	Vorhergehenden Satz löschen.
C-x e	call-last-kbd-macro	Zuletzt definiertes Makro ausführen.
C-x h	mark-whole-buffer	Buffer markieren.
C-x i	insert-file	Datei an der Cursor-Position einfügen.
C-x k	kill-buffer	Buffer löschen.
C-x o	other-window	Zum anderen Fenster wechseln.
C-x q	kbd-macro-query	Abfrage in Makro-Definition einfügen.
C-x s	save-some-buffers	Für jeden Buffer fragen, ob er gespeichert werden soll.
C-x u	advertised-undo	Letzten Editiervorgang rückgängig machen (kann auch wiederholt verwendet werden).
C-y	yank	Gelöschten Text zurückholen.
C-z	suspend-emacs	Emacs unterbrechen (kann mit **exit** oder **fg** wieder aktiviert werden).

Tastenkombinationen mit Meta

Bindung	Befehl	Aktion
M- – M-c	negatives Argument; capitalize-word	Ersten Buchstaben des vorhergehenden Wortes groß schreiben.
M- – M-l	negatives Argument; downcase-word	Ganzes vorhergehendes Wort in Kleinbuchstaben schreiben.
M- – M-u	negatives Argument; upcase-word	Ganzes vorhergehendes Wort in Großbuchstaben schreiben.
M-$	spell-word	Rechtschreibung des Wortes nach dem Cursor überprüfen.
M-<	beginning-of-buffer	Cursor zum Bufferanfang bewegen.
M->	end-of-buffer	Cursor zum Bufferende bewegen.
M-{	backward-paragraph	Cursor einen Absatz rückwärts bewegen.
M-}	forward-paragraph	Cursor einen Absatz vorwärts bewegen.
M-^	delete-indentation	Aneinanderhängen dieser und der vorhergehenden Zeile.
M-*n*	digit-argument	Nächsten Befehl *n*-mal ausführen.
M-*n* C-x e	digit argument und call-last-kbd-macro	Zuletzt definiertes Makro *n*-mal ausführen.
M-a	backward-sentence	Cursor einen Satz rückwärts bewegen.
M-b	backward-word	Cursor ein Wort rückwärts bewegen.
M-C-\	indent-region	Region so einrücken, daß sie zur ersten Zeile der Region paßt.
M-C-c	exit-recursive-edit	Rekursives Editieren beenden.
M-C-o	split-line	Zeile an der Cursor-Position aufbrechen; bis zur Cursorspalte einrücken.

Bindung	Befehl	Aktion
M-C-r	isearch-backward-regexp	Rückwärts gerichtete inkrementelle Suche nach regulären Ausdrücken.
M-C-s	isearch-forward-regexp	Vorwärts gerichtete inkrementelle Suche nach regulären Ausdrücken.
M-C-v	scroll-other-window	Anderes Fenster scrollen.
M-c	capitalize-word	Ersten Buchstaben eines Wortes groß schreiben.
M-d	kill-word	Wort unter dem Cursor löschen.
M-Entf	backward-kill-word	Vorhergehendes Wort löschen.
M-e	forward-sentence	Cursor einen Satz vorwärts bewegen.
M-f	forward-word	Cursor ein Wort vorwärts bewegen.
M-g	fill-region	Region neu formatieren.
M-h	mark-paragraph	Absatz markieren.
M-k	kill-sentence	Satz unter dem Cursor löschen.
M-l	downcase-word	Ganzes Wort in Kleinbuchstaben schreiben.
M-m	back-to-indentation	Cursor auf das erste Zeichen der Zeile setzen.
M-q	fill-paragraph	Absatz neu formatieren.
M-s	center-line	Aktuelle Zeile zentrieren.
M-t	transpose-words	Zwei Wörter vertauschen.
M-u	upcase-word	Ganzes Wort in Großbuchstaben schreiben.
M-v	scroll-down	Cursor eine Bildschirmseite nach oben (!) bewegen.
M-x	(keiner)	Befehl über den Befehlsnamen aufrufen.

Befehlsübersicht nach Befehlsnamen sortiert

Die nachstehenden Emacs-Befehle sind in alphabetischer Reihenfolge geordnet. Mit M-x kann ein Befehl über seinen Namen aufgerufen werden. Die Tabellen geben jeweils den Befehlsnamen, die Tastenkombination und die ausgeführte Aktion an. C- bezeichnet die Control-, M- die Meta-Taste.

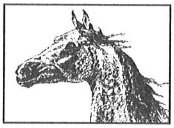
Befehl	Bindung	Aktion
macroname	(keine)	Gespeichertes Makro ausführen.
abbrev-mode	(keine)	Wort-Abkürzungsmodus starten oder beenden.
advertised-undo	C-x u	Letzten Editiervorgang rückgängig machen (kann auch wiederholt verwendet werden).
apropos	(keine)	Welche Befehle, Funktionen und Variablen gibt es zu diesem Stichwort?
back-to-indentation	M-m	Cursor auf das erste Zeichen der Zeile setzen.
backward-char	C-b	Cursor ein Zeichen rückwärts (nach links) bewegen.
backward-delete-char	Entf	Zeichen links vom Cursor löschen.
backward-kill-paragraph	(keine)	Vorhergehenden Absatz löschen.
backward-kill-sentence	C-x Entf	Vorhergehenden Satz löschen.
backward-kill-word	C-c C-w	Vorhergehendes Wort löschen.
backward-kill-word	M-Entf	Vorhergehendes Wort löschen.
backward-page	C-x [	Cursor eine Seite rückwärts bewegen.
backward-paragraph	M-{	Cursor einen Absatz rückwärts bewegen.
backward-sentence	M-a	Cursor einen Satz rückwärts bewegen.
backward-word	M-b	Cursor ein Wort rückwärts bewegen.
beginning-of-buffer	M-<	Cursor zum Bufferanfang bewegen.
beginning-of-line	C-a	Cursor an den Zeilenanfang bewegen.
call-last-kbd-macro	C-x e	Zuletzt definiertes Makro ausführen.

Befehl	Bindung	Aktion
capitalize-region	(keine)	Alle Initialbuchstaben der Region groß schreiben.
capitalize-word	M-c	Ersten Buchstaben eines Wortes groß schreiben.
negatives Argument; capitalize-word	M- – M-c	Ersten Buchstaben des vorhergehenden Wortes groß schreiben.
center-line	M-s	Aktuelle Zeile zentrieren.
center-paragraph	(keine)	Aktuellen Absatz zentrieren.
center-region	(keine)	Aktuelle Region zentrieren.
command-apropos	C-h a	Welche Befehle gibt es zu diesem Stichwort?
compare-windows	(keine)	Zwei Buffer vergleichen; ersten Unterschied anzeigen.
delete-char	C-d	Zeichen unter dem Cursor löschen.
delete-indentation	M-^	Diese und die vorhergehende Zeile zu einer zusammenfassen.
delete-other-windows	C-x 1	Alle Fenster außer dem aktuellen löschen.
delete-window	C-x 0	Aktuelles Fenster löschen.
delete-windows-on	(keine)	Alle zu einem bestimmten Buffer gehörenden Fenster löschen.
describe-bindings	C-h b	Welche Tastenbindungen gibt es für diesen Buffer?
describe-copying	C-h C-c	Anzeige der Emacs General Public License.
describe-distribution	C-h C-d	Informationen, wie Emacs von der FSF bestellt werden kann.
describe-function	C-h f	Was macht diese Funktion?
describe-key	C-h k	Welchen Befehl führt diese Tastenkombination aus und was macht er?
describe-key-briefly	C-h c	Welcher Befehl wird durch diese Tastenkombination ausgeführt?
describe-mode	C-h m	Ausgabe einer Beschreibung des aktuellen Modus.
describe-no-warranty	C-h C-w	Informationen über die (nicht vorhandene) Garantie von Emacs.
describe-syntax	C-h s	Wie lautet die Syntaxtabelle dieses Buffers?
describe-variable	C-h v	Was bedeutet diese Variable, und wie lautet ihr Wert?
digit-argument	M-*n*	Nächsten Befehl *n*-mal ausführen.
downcase-region	C-x C-l	Ganze Region in Kleinbuchstaben schreiben.
downcase-word	M-l	Ganzes Wort in Kleinbuchstaben schreiben.
negatives Argument; downcase-word	M- – M-l	Ganzes vorheriges Wort in Kleinbuchstaben schreiben.
edit-abbrevs	(keine)	Wort-Abkürzungen editieren.
end-kbd-macro	C-x)	Makro-Definition beenden.
end-of-buffer	M->	Cursor zum Bufferende bewegen.
end-of-line	C-e	Cursor an das Zeilenende bewegen.
enlarge-window	C-x ^	Fenster vergrößern.
enlarge-window-horizontally	C-x }	Fenster verbreitern.
exchange-point-and-mark	C-x C-x	Positionen des Cursors und der Markierung vertauschen.
exit-recursive-edit	M-C-c	Rekursives Editieren beenden.
fill-individual-paragraphs	(keine)	Eingerückte Absätze neu formatieren; Einrückungen beibehalten.
fill-paragraph	M-q	Absatz neu formatieren.
fill-region	M-g	Region neu formatieren.
find-alternate-file	C-x C-v	eine andere Datei einlesen; ersetzt die mit C-x C-f eingelesene.
find-file	C-x C-f	Datei einlesen.
find-file-other-window	C-x 4 f	Datei laden und im anderen Fenster anzeigen.

Befehl	Bindung	Aktion
forward-char	C-f	Cursor ein Zeichen vorwärts (nach rechts) bewegen.
forward-page	C-x]	Cursor eine Seite vorwärts bewegen.
forward-paragraph	M-}	Cursor einen Absatz vorwärts bewegen.
forward-sentence	M-e	Cursor einen Satz vorwärts bewegen.
forward-word	M-f	Cursor ein Wort vorwärts bewegen.
goto-char	(keine)	Cursor zum Zeichen *n* bewegen.
goto-line	(keine)	Cursor zur Zeile *n* bewegen.
help-command	C-h	Aufrufen des Online Hilfe-Systems.
help-with-tutorial	C-h t	Starten des Emacs-Lernprogramms.
indent-region	M-C-\	Region so einrücken, daß sie zur ersten Zeile der Region paßt.
indented-text-mode	(keine)	Major-Mode. Jeder Tabulator definiert eine neue Einrückung für die folgenden Zeilen.
info	C-h i	Starten des Leseprogramms für Info-Dateien.
insert-file	C-x i	Datei an der Cursor-Position einfügen.
insert-last-keyboard-macro	(keine)	Benanntes Makro in eine Datei einfügen.
interrupt-shell-subjob	C-c C-c	Aktuellen Job abbrechen.
inverse-add-global-abbrev	C-x -	Tlobale Abkürzung und ihre Definition eingeben.
inverse-add-local-abbrev	C-x C-h	Lokale Abkürzung und ihre Definition eingeben.
isearch-backward	C-r	Rückwärts gerichtete inkrementelle Suche beginnen oder wiederholen.
isearch-backward-regexp	M-C-r	Rückwärts gerichtete inkrementelle Suche nach regulären Ausdrücken.
isearch-forward	C-s	Vorwärts gerichtete inkrementelle Suche beginnen oder wiederholen.
isearch-forward-regexp	M-C-s	Vorwärts gerichtete inkrementelle Suche nach regulären Ausdrücken.
kbd-macro-query	C-x q	Abfrage in Makro-Definition einfügen.
keyboard-quit	C-g	Aktuellen Befehl abbrechen.
kill-all-abbrevs	(keine)	Alle Wort-Abkürzungen für die Dauer dieser Sitzung löschen.
kill-buffer	C-x k	Buffer löschen.
kill-line	C-k	Alles von der Cursor-Position bis zum Zeilenende löschen.
kill-paragraph	(keine)	Alles von der Cursor-Position bis zum Absatzende löschen.
kill-region	C-w	Markierten Bereich löschen (siehe nächsten Abschnitt)
kill-sentence	M-k	Satz unter dem Cursor löschen.
kill-shell-input	C-c C-u	Aktuelle Zeile löschen.
kill-some-buffers	(keine)	Für jeden Buffer fragen, ob er gelöscht werden soll.
kill-word	M-d	Wort unter dem Cursor löschen.
list-abbrevs	(keine)	Wort-Abkürzungen anzeigen.
list-buffers	C-x C-b	Bufferliste anzeigen.
load-file	(keine)	Gespeicherte Makros laden.
mark-page	C-x C-p	Seite markieren.
mark-paragraph	M-h	Absatz markieren.
mark-whole-buffer	C-x h	Buffer markieren.
name-last-kbd-macro	(keine)	Zuletzt angelegtes Makro benennen (vor dem Abspeichern).
next-line	C-n	Cursor eine Zeile vorwärts (nach unten) bewegen.
other-window	C-x o	Zum anderen Fenster wechseln.

Befehl	Bindung	Aktion
previous-line	C-p	Cursor eine Zeile rückwärts (nach oben) bewegen.
query-replace-regexp	(keine)	Suchen und Ersetzen mit einem regulären Ausdruck.
recenter	C-l	Bildschirm so neu zeichnen, daß die aktuelle Zeile in der Bildschirmmitte liegt.
rename-buffer	(keine)	Buffer umbenennen.
revert-buffer	(keine)	Buffer wieder in den Zustand versetzen, in dem er sich befand, als die Datei das letzte Mal gespeichert wurde (auch automatisch).
save-buffer	C-x C-s	Datei speichern.
save-buffers-kill-emacs	C-x C-c	Emacs beenden.
save-some-buffers	C-x s	Für jeden Buffer fragen, ob er gespeichert werden soll.
scroll-down	M-v	Cursor eine Bildschirmseite nach oben (!) bewegen.
scroll-left	C-x <	Fenster nach links scrollen.
scroll-other-window	M-C-v	Anderes Fenster scrollen.
scroll-right	C-x >	Fenster nach rechts scrollen.
scroll-up	C-v	Cursor eine Bildschirmseite nach unten (!) bewegen.
set-fill-prefix	C-x .	Jeder Zeile des Absatzes vom Zeilenanfang bis zur aktuellen Cursorspalte Zeichen voranstellen. Wird durch Ausführung dieses Befehls mit dem Cursor in Spalte 1 gelöscht.
set-mark-command	C-@ or C-SPACE	Anfang (oder Ende) einer Region markieren.
shell-send-eof	C-c C-d	Dateiendezeichen.
shrink-window	(keine)	Fenster verkleinern.
shrink-window-horizontally	C-x {	Fenster schmaler machen.
spell-buffer	(keine)	Rechtschreibung des aktuellen Buffers überprüfen.
spell-region	(keine)	Rechtschreibung der aktuellen Region überprüfen.
spell-string	(keine)	Rechtschreibung einer im Minibuffer eingegebenen Zeichenkette überprüfen
spell-word	M-$	Rechtschreibung des Wortes nach dem Cursor überprüfen.
split-line	M-C-o	Zeile an der Cursor-Position aufbrechen; bis zur Cursorspalte einrücken.
split-window-horizontally	C-x 2	Fenster horizontal in zwei aufteilen.
split-window-vertically	C-x 5	Fenster vertikal in zwei aufteilen.
start-kbd-macro	C-x (	Makro-Definition anfangen.
stop-shell-subjob	C-c C-z	Aktuellen Job unterbrechen.
suspend-emacs	C-z	Emacs unterbrechen (kann mit **exit** oder **fg** wieder aktiviert werden).
switch-to-buffer	C-x b	Zum angegebenen Buffer wechseln.
switch-to-buffer-other-window	C-x 4 b	Buffer im anderen Fenster auswählen.
text-mode	(keine)	Indented-text-mode verlassen und zum normalen Text-Modus zurückkehren.
transpose-chars	C-t	Zwei Zeichen vertauschen.
transpose-lines	C-x C-t	Zwei Zeilen vertauschen.
transpose-paragraphs	(keine)	Zwei Absätze vertauschen.
transpose-sentences	(keine)	Zwei Sätze vertauschen.
transpose-words	M-t	Zwei Wörter vertauschen.
unexpand-abbrev	(keine)	Letzte Wort-Abkürzung rückgängig machen.
universal-argument	C-u *n*	Nächsten Befehl *n*-mal ausführen.
upcase-region	C-x C-u	Ganze Region in Großbuchstaben schreiben.
upcase-word	M-u	Ganzes Wort in Großbuchstaben schreiben.

Befehl	Bindung	Aktion
negatives Argument; upcase-word	M- – M-u	Ganzes vorhergehendes Wort in Großbuchstaben schreiben.
view-emacs-news	C-h n	Anzeigen von Neuigkeiten über neue Emacs-Versionen.
view-lossage	C-h l	Was waren die letzten hundert eingegebenen Zeichen?
where-is	C-h w	Wie lautet die Tastenbindung für diesen Befehl?
write-abbrev-file	(keine)	Wort-Abkürzungsdatei abspeichern.
write-file	C-x C-w	Bufferinhalt in eine Datei schreiben.
yank	C-y	Gelöschten Text zurückholen.



KAPITEL 7

Der Editor vi

Auf Linux-Systemen steht üblicherweise eine erweiterte Version des **vi** namens **vim** oder **nvi** zur Verfügung. In diesem Kapitel werden die folgenden Punkte behandelt:

- Überblick über das Arbeiten mit dem **vi**
- Cursorbewegungen
- Editierbefehle
- Speichern und Beenden
- Bearbeitung mehrerer Dateien
- Shell-Interaktion
- Makros
- Diverse Befehle
- Alphabetische Tastenübersicht
- **vi**-Einstellungen

Überblick über das Arbeiten mit dem vi

In diesem Abschnitt finden Sie die folgenden Punkte:

- Syntax der Kommandozeile
- **vi**-Modi
- Syntax der **vi**-Befehle
- Befehle auf der Statuszeile

Mehr Informationen über **vi** finden Sie in *Textbearbeitung mit dem vi-Editor* von Linda Lamb.

Syntax der Kommandozeile

Die drei gebräuchlichsten Befehle zum Starten einer **vi**-Sitzung sind:

vi *Datei(en)*
vi +*n* *Datei(en)*
vi +/*Suchmuster* *Datei(en)*

Sie können eine *Datei* zum Editieren öffnen und optional eine Zeilennummer oder ein Muster angeben, wenn **vi** an einer bestimmten Stelle beginnen soll. Wenn keine Datei angegeben wird, startet **vi** mit einem leeren Buffer.

Befehlsmodus

Wenn eine Datei geöffnet wurde, befinden Sie sich im Befehlsmodus. Hier können Sie folgende Operationen ausführen:

- Zum Einfügemodus wechseln
- Editierbefehle eingeben
- Den Cursor bewegen
- **ex**-Befehle aufrufen
- Eine Linux-Shell starten
- Die aktuelle Version der Datei abspeichern und **vi** beenden.

Einfügemodus

Im Einfügemodus können Sie neuen Text eingeben. Durch Drücken der ESCAPE-Taste verlassen Sie den Einfügemodus und kehren zum Befehlsmodus zurück. Mit den folgenden Befehlen können Sie den Einfügemodus aktivieren:

`a`	Hinter dem Cursor anhängen.
`A`	Am Ende der Zeile anhängen.
`c`	Austausch-Operation starten.
`C`	Text bis zum Zeilenende austauschen.
`i`	Vor dem Cursor einfügen.
`I`	Am Zeilenanfang einfügen.
`o`	Einfügen einer Zeile unter der aktuellen.
`O`	Einfügen einer Zeile über der aktuellen.
`R`	Text überschreiben.
`s`	Ein Zeichen austauschen.
`S`	Eine Zeile austauschen.

Syntax der vi-Befehle

vi-Befehle haben allgemein die folgende Form:

[*n*] *Operator* [*m*] *Objekt*

Die grundlegenden Editieroperatoren sind:

`c`	Änderung.
`d`	Löschen.
`y`	Operationen mit der Zwischenablage.

Wenn das Objekt der Operation die aktuelle Zeile ist, ist der Buchstabe für das Objekt derselbe wie der des Operators: **cc**, **dd**, **yy**. Anderenfalls arbeiten die Operatoren auf Objekten, die durch Befehle zur Cursorbewegung oder zum Mustervergleich bestimmt werden. *n* und *m* geben an, wie oft bzw. auf wie vielen Objekten eine Operation ausgeführt werden soll. Wenn sowohl *n* als auch *m* angegeben werden, wird die Operation $n \times m$-mal ausgeführt.

Ein Objekt kann einer der folgenden Textblöcke sein:

Wort	Alle Zeichen bis zu einem Leer- oder Satzzeichen. Wenn der Objektbuchstabe groß geschrieben wird, werden nur Leerzeichen als Trennzeichen erkannt.
Satz	Alle Zeichen bis zu einem `.`, `!` oder `?`, gefolgt von zwei Leerzeichen.
Absatz	Alle Zeichen bis zur nächsten leeren Zeile oder einem Absatzmakro, das durch die **para=**-Option definiert wird.
Abschnitt	Alle Zeichen bis zur nächsten Abschnittsüberschrift, die durch die **sect=**-Option definiert wird.

Beispiele

`2cw`	Vertauschen der nächsten beiden Wörter.
`d}`	Alles bis zum Beginn des nächsten Absatzes löschen.
`d^`	Alles zurück bis zum Zeilenanfang löschen.
`5yy`	Kopieren der nächsten fünf Zeilen in die Zwischenablage.
`y]]`	Alles bis zum nächsten Abschnittsbeginn kopieren.

Befehle auf der Statuszeile

Die meisten Befehle werden während der Eingabe nicht auf dem Bildschirm angezeigt. Bei den folgenden Befehlen wird die Eingabe aber in der Statuszeile, der letzten Zeile des Bildschirms, wiedergegeben:

`/`	Vorwärts gerichtete Suche nach einem Muster.
`?`	Rückwärts gerichtete Suche nach einem Muster.
`:`	Aufrufen eines **ex**-Befehls.
`!`	Aufrufen eines Shell-Befehls, der als Eingabe ein Objekt des aktuellen Buffers enthält und dieses durch die Befehlsausgabe ersetzt.

Befehle, die auf der Statuszeile eingegeben werden, werden mit der RETURN-Taste abgeschlossen. Außerdem werden auf der Statuszeile Fehlermeldungen und die Ausgabe des **CTRL-G**-Befehls angezeigt.

Cursorbewegungen

Wird vor dem jeweiligen Befehl eine Zahl eingegeben, wird der Befehl entsprechend oft wiederholt. Cursorbewegungen können auch als Objekte für Austausch-, Lösch- und Kopieroperationen dienen.

Zeichenweise

`h, j, k, l`	Nach links, nach unten, nach oben, nach rechts (←, ↓, ↑, →)
`Leertaste`	Nach rechts

Text

`w, W, b, B`	Wortweise vorwärts und rückwärts
`e, E`	Zum Wortende
`), (`	Zum Anfang des nächsten bzw. aktuellen Satzes
`}, {`	Zum Anfang des nächsten bzw. aktuellen Absatzes
`]], [[`	Zum Anfang des nächsten bzw. aktuellen Abschnitts

Zeilenweise

`0, $`	Erstes bzw. letztes Zeichen der aktuellen Zeile
`^`	Erstes nicht-leeres Zeichen der aktuellen Zeile
`+, -`	Erstes Zeichen der nächsten bzw. vorhergehenden Zeile
`RETURN`	Erstes Zeichen der nächsten Zeile
n`|`	Spalte *n* der aktuellen Zeile
`H`	Erste Bildschirmzeile
`M`	Mittlere Bildschirmzeile
`L`	Letzte Bildschirmzeile
n`H`	*n*-te Bildschirmzeile
n`L`	*n*-te Bildschirmzeile von unten

Bildschirmweise

`CTRL-F, CTRL-B`	Einen Bildschirm vorwärts bzw. rückwärts scrollen.
`CTRL-D, CTRL-U`	Einen halben Bildschirm vorwärts bzw. rückwärts scrollen.
`CTRL-E, CTRL-Y`	Anzeigen einer weiteren Zeile am Bildschirmende bzw. -anfang.
`z` RETURN	Aktuelle Zeile am Bildschirmanfang anzeigen.
`z .`	Aktuelle Zeile in der Bildschirmmitte anzeigen.
`z -`	Aktuelle Zeile am Bildschirmende anzeigen.
`CTRL-L, CTRL-R`	Bildschirm neu zeichnen (ohne zu scrollen).

Suchen

`/`*`Suchmuster`*	Vorwärts gerichtete Suche nach *`Suchmuster`*.
`/`	Vorwärts gerichtete Suche wiederholen.
`/`*`Suchmuster`*`/+`*`n`*	Sprung zur *n*-ten Zeile nach *`Suchmuster`*.
`?`*`Suchmuster`*	Rückwärts gerichtete Suche nach *`Suchmuster`*.
`?`	Rückwärts gerichtete Suche wiederholen.

`?`*Suchmuster*`?-`*n*	Sprung zur *n*-ten Zeile vor *Suchmuster*.
`n`	Letzte Suchoperation wiederholen.
`N`	Letzte Suchoperation in umgekehrter Richtung wiederholen.
`%`	Suchen der passenden runden, eckigen oder geschweiften Klammer.
`f`*x*	Cursor auf das nächste *x* der aktuellen Zeile positionieren.
`F`*x*	Cursor auf das letzte *x* der aktuellen Zeile positionieren.
`t`*x*	Cursor ein Zeichen vor dem nächsten *x* der aktuellen Zeile positionieren.
`T`*x*	Cursor ein Zeichen hinter dem letzten *x* der aktuellen Zeile positionieren.
`,`	Richtung der letzten Suche mit **f**, **F**, **t**, oder **T** umdrehen.
`;`	Letzte Zeichensuche (**f**, **F**, **t**, oder **T**) wiederholen.

Zeilennumerierung

`CTRL-G`	Nummer der aktuellen Zeile anzeigen.
n`G`	Sprung zur Zeile *n*.
`G`	Sprung zur letzten Zeile der Datei.
`:`*n*	Sprung zur Zeile *n*.

Markieren von Positionen

`m`*x*	Aktuelle Position mit dem Zeichen *x* markieren.
`` ` ``*x*	(Rückwärtiges Anführungszeichen) Cursor auf Stelle mit Markierung *x* positionieren.
`'`*x*	(Apostroph) Cursor auf den Anfang der Zeile, welche die Markierung *x* enthält, positionieren.
` `` `	(Rückwärtige Anführungszeichen) Rücksprung zur letzten Markierung (oder zu der Stelle, an der eine Suche begonnen wurde).
`''`	(Apostrophe) Wie oben, aber Rücksprung an den Start der Zeile.

Editierbefehle

Zur Erinnerung: **c**, **d** und **y** sind die grundlegenden Editieroperatoren.

Neuen Text einfügen

`a`	Hinter dem Cursor einfügen.
`A`	Am Ende der Zeile anhängen.
`i`	Vor dem Cursor einfügen.
`I`	Am Anfang der Zeile einfügen.
`o`	Einfügen einer neuen Zeile oberhalb des Cursors.
`O`	Einfügen einer neuen Zeile unterhalb des Cursors.
`ESC`	Einfügemodus beenden.
`TAB`	Tabulatorzeichen einfügen.
`Rueckschritt`	Ein Zeichen zurück bewegen.
`RETURN`	Zum ersten Zeichen der nächsten Zeile springen.
`CTRL-J`	Eine Zeile nach unten bewegen.
`CTRL-I`	Tabulatorzeichen einfügen.
`CTRL-T`	Zur nächsten Tabulatorposition springen.
`CTRL-D`	Zur vorhergehenden Tabulatorposition springen.
`CTRL-H`	Ein Zeichen rückwärts bewegen.

`CTRL-U`	Aktuelle Zeile löschen.
`CTRL-V`	Nächstes Zeichen uninterpretiert einfügen.
`CTRL-W`	Ein Wort zurück bewegen.

Die letzten vier CTRL-Befehle werden durch die **stty**-Einstellungen bestimmt. Auf Ihrem Terminal können auch andere Einstellungen vorkommen.

Text verändern und löschen

`cw`	Wort ändern.
`cc`	Zeile ändern.
`C`	Text von der aktuellen Position bis zum Zeilenende ändern.
`dd`	Aktuelle Zeile löschen.
n`dd`	*n* Zeilen löschen.
`D`	Bis zum Zeilenende löschen.
`dw`	Wort löschen.
`d}`	Bis zum nächsten Absatz löschen.
`d^`	Bis zum Zeilenanfang rückwärts löschen.
`d/`*Muster*	Bis zum ersten Vorkommen des Musters löschen.
`dn`	Bis zum letzten Vorkommen des Musters löschen.
`df`*a*	Alle Zeichen bis zum nächsten *a* auf der aktuellen Zeile (einschließlich) löschen.
`dt`*a*	Alle Zeichen bis zum nächsten *a* auf der aktuellen Zeile (ausschließlich) löschen.
`dL`	Bis zum Bildschirmende löschen.
`dG`	Bis zum Dateiende löschen.
`p`	Zuletzt gelöschten Text hinter dem Cursor einfügen.
`P`	Zuletzt gelöschten Text vor dem Cursor einfügen.
`r`*x*	Zeichen durch *x* ersetzen.
`R`*Text*	*Text* ab der Cursor-Position ersetzen.
`s`	Ein Zeichen austauschen.
n`s`	*n* Zeichen austauschen.
`S`	Ganze Zeile austauschen.
`u`	Letzte Änderung rückgängig machen.
`U`	Aktuelle Zeile restaurieren.
`x`	Zeichen unter dem Cursor löschen.
`X`	Ein Zeichen nach links löschen.
n`X`	Letzte *n* Zeichen löschen.
`.`	Letzte Änderung wiederholen.
`~`	Groß-/Kleinschreibung vertauschen.

Kopieren und Verschieben

`Y`	Aktuelle Zeile in einen neuen Buffer kopieren.
`yy`	Aktuelle Zeile kopieren.
`"`*x*`yy`	Aktuelle Zeile in Buffer *x* kopieren.
`"`*x*`d`	Aktuelle Zeile in Buffer *x* verschieben.
`"`*X*`d`	Aktuelle Zeile an Buffer *x* anhängen und löschen.
`"`*x*`p`	Inhalt von Buffer *x* einfügen.

`y]]`	Alles bis zur nächsten Abschnittsüberschrift kopieren.
`ye`	Alles bis zum Wortende kopieren.

Speichern und Beenden

Wegschreiben einer Datei bedeutet hier das Abspeichern der Änderungen und das Aktualisieren des Zeitstempels der Datei.

`ZZ`	**vi** beenden, Datei nur wegschreiben, wenn Änderungen gemacht worden sind.
`:x`	Wie **ZZ**.
`:wq`	Wegschreiben und beenden.
`:w`	Wegschreiben.
`:w` *`Datei`*	Kopie des Buffers nach *`Datei`* schreiben.
`:`*`n1`*`,`*`n2`*`w` *`Datei`*	Zeilen *`n1`* bis *`n2`* in die neue *`Datei`* schreiben.
`:`*`n1`*`,`*`n2`*`w >>` *`Datei`*	Zeilen *`n1`* bis *`n2`* an die existierende *`Datei`* anhängen.
`:w!`	Datei wegschreiben, dabei Schreibschutz (wenn möglich) nicht beachten.
`:w!` *`Datei`*	Datei *`Datei`* mit dem aktuellen Buffer überschreiben.
`:w %.`*`neu`*	Den aktuellen Buffer namens *`Datei`* als *`Datei.neu`* speichern.
`:q`	Bearbeiten der Datei beenden.
`:q!`	Bearbeiten der Datei beenden, dabei Änderungen verwerfen.
`Q`	**vi** beenden und **ex** aufrufen.
`:vi`	Rückkehr zum **vi** nach einem Q-Befehl.
`:e` *`Datei2`*	*`Datei2`* editieren, ohne **vi** zu verlassen.
`:n`	Nächste Datei editieren.
`:e!`	Zustand beim letzten Speichern der Datei restaurieren.
`:e#`	Andere Datei editieren.
`%`	Aktueller Dateiname.
`#`	Alternativer Dateiname.

Bearbeiten mehrerer Dateien

`:e` *`Datei`*	Andere *`Datei`* bearbeiten; aktuelle Datei wird alternative Datei.
`:e!`	Letzte gespeicherte Version der Datei zurückholen.
`:e +` *`Datei`*	Editieren am Ende von *`Datei`* anfangen.
`:e +`*`n Datei`*	*`Datei`* an Zeile *`n`* öffnen.
`:e #`	Auf die vorige Position in der alternativen Datei springen.
`:ta` *`Tag`*	Datei an der Stelle *`Tag`* editieren.
`:n`	Nächste Datei editieren.
`:n!`	Nächste Datei editieren, Änderungen ggf. verwerfen.
`:n` *`Dateien`*	Neue Dateiliste *`Dateien`* angeben.
`CTRL-G`	Aktuellen Dateinamen und Zeilennummer anzeigen.
`:args`	Alle zu editierenden Dateien anzeigen.
`:rew`	Zur ersten zu bearbeitenden Datei zurückspringen.

Shell-Interaktion

`:r` *`Datei`*	Inhalt der *`Datei`* hinter dem Cursor einfügen.
`:r !`*`Befehl`*	Ausgabe des *`Befehls`* hinter der aktuellen Zeile einfügen.
`:`*`n`*`r !`*`Befehl`*	Wie zuvor, aber hinter Zeile *`n`* einfügen (0 bedeutet am Dateianfang).
`:!`*`Befehl`*	*`Befehl`* ausführen.
`!`*`Objekt Befehl`*	*`Befehl`* mit dem Buffer *`Objekt`* als Eingabe starten und das Objekt durch die Ausgabe ersetzen.
`:`*`n1,n2`*`!` *`Befehl`*	*`Befehl`* mit den Zeilen *`n1`* – *`n2`* als Eingabe starten und die Zeilen durch die Ausgabe ersetzen.
`n``!!`*`Befehl`*	*`Befehl`* mit den nächsten *`n`* Zeilen als Eingabe starten und die Zeilen durch die Ausgabe ersetzen.
`!!`	Letzten Shell-Befehl wiederholen.
`:sh`	Subshell erzeugen; Rückkehr zur gerade bearbeiteten Datei mit *EOF (CTRL-D)*.
`CTRL-Z`	Editor unterbrechen, weitermachen mit **fg**.
`:so` *`Datei`*	Befehle aus *`Datei`* lesen und ausführen.

Makros

`:ab` *`Abk Text`*	*`Abk`* als Abkürzung für *`Text`* verwenden.
`:unab` *`Abk`*	Abkürzung für *`Abk`* löschen.
`:ab`	Abkürzungen anzeigen.
`:map` *`c String`*	Zeichen *`c`* auf die Befehlsfolge *`String`* abbilden.
`:unmap` *`c`*	Abbildung für das Zeichen *`c`* löschen.
`:map`	Abgebildete Zeichen anzeigen.
`:map!` *`c String`*	Zeichen *`c`* auf die Zeichenfolge *`String`* im Einfügemodus abbilden.
`:unmap!` *`c`*	Abbildung des Zeichens *`c`* für den Einfügemodus abschalten (evtl. muß das Zeichen mit **CTRL-V** gequotet werden).
`:map!`	Zeichen, die im Einfügemodus abgebildet werden, anzeigen.

Die folgenden Zeichen werden im Befehlsmodus nicht verwendet und können auf benutzerdefinierte Befehle abgebildet werden.

Buchstaben:	`g K q V v`
Control-Kombinationen:	`^A ^K ^O ^T ^W ^X`
Symbole:	`_ * \ =`

(Anmerkung: Das Zeichen = wird vom **vi** im Lisp-Modus verwendet.)

Diverse Befehle

`J`	Zwei Zeilen zusammenfassen.
`:j!`	Zwei Zeilen zusammenfassen, dabei Leerzeichen erhalten.
`<<`	Die aktuelle Zeile eine Verschiebebreite nach links verschieben (Default ist acht Zeichen).
`>>`	Die aktuelle Zeile eine Verschiebebreite nach rechts verschieben (Default ist acht Zeichen).
`>}`	Bis zum Absatzende nach rechts verschieben.

`<%`	Bis zur dazugehörigen runden, eckigen, geschweiften etc. Klammer nach links verschieben (der Cursor muß auf dem entsprechenden Zeichen stehen).

Alphabetische Tastenliste

Aus Platzgründen werden Control-Kombinationen mit ^ gekennzeichnet.

`a`	Text hinter dem Cursor einfügen.
`A`	Text am Ende der Zeile anhängen.
`^A`	Unbenutzt.
`b`	Zurück zum Wortanfang auf der aktuellen Zeile.
`B`	Zurück zum Wortanfang, Satzzeichen nicht berücksichtigen.
`^B`	Eine Fenstergröße zurückscrollen.
`c`	Austausch-Operator.
`C`	Bis zum Zeilenende ersetzen.
`^C`	Unbenutzt im Befehlsmodus; beendet den Einfügemodus.
`d`	Lösch-Operator.
`D`	Bis zum Ende der aktuellen Zeile löschen.
`^D`	Halbe Fenstergröße herunterscrollen; im Einfügemodus Einrückung auf **shiftwidth** aufheben, wenn **autoindent** eingeschaltet ist.
`e`	Zum Wortende springen.
`E`	Zum Wortende springen, Satzzeichen nicht berücksichtigen.
`^E`	Eine weitere Zeile am Ende des Fensters anzeigen.
`f`	Nächstes Zeichen nach vorn auf der aktuellen Zeile suchen.
`F`	Nächstes Zeichen nach hinten auf der aktuellen Zeile suchen.
`^F`	Ein Fenster nach vorn scrollen.
`g`	Unbenutzt.
`G`	Zur angegebenen Zeile oder zum Zeilenende springen.
`^G`	Informationen über die Datei in der Statuszeile ausgeben.
`h`	Cursor ein Zeichen nach links bewegen.
`H`	Cursor auf das erste Zeichen der Datei setzen.
`^H`	Cursor ein Zeichen nach links bewegen; im Einfügemodus Rückschritt-Taste.
`i`	Text vor dem Cursor einfügen.
`I`	Text vor dem ersten nicht-leeren Zeichen der aktuellen Zeile einfügen.
`^I`	Unbenutzt im Befehlsmodus; im Einfügemodus Tabulatortaste.
`j`	Cursor eine Zeile nach unten bewegen.
`J`	Zwei Zeilen zusammenfassen.
`^J`	Cursor eine Zeile nach unten bewegen (auch im Einfügemodus).
`k`	Cursor eine Zeile nach oben bewegen.
`K`	Unbenutzt.
`^K`	Unbenutzt.
`l`	Cursor ein Zeichen nach rechts bewegen.
`L`	Cursor auf das letzte sichtbare Zeichen bewegen.
`^L`	Bildschirm neu zeichnen.

`m`	Aktuelle Cursorposition im Register (a-z) festhalten.
`M`	Cursor auf die mittlere sichtbare Zeile positionieren.
`^M`	Zeilensprung.
`n`	Letzten Suchbefehl wiederholen.
`N`	Letzten Suchbefehl in umgekehrter Richtung wiederholen.
`^N`	Cursor eine Zeile nach unten bewegen.
`o`	Neue Zeile unter der aktuellen einfügen.
`O`	Neue Zeile über der aktuellen einfügen.
`^O`	Unbenutzt.
`p`	Gelöschten oder kopierten Text hinter oder unter dem Cursor einfügen.
`P`	Gelöschten oder kopierten Text vor oder über dem Cursor einfügen.
`^P`	Cursor ein Zeichen nach oben bewegen.
`q`	Unbenutzt.
`Q`	**vi** beenden und **ex** aufrufen.
`^Q`	Unbenutzt. (Auf manchen Terminals Wiederaufnahme der Datenübertragung.)
`r`	Zeichen unter dem Cursor durch das nächste eingegebene Zeichen ersetzen.
`R`	Zeichen ersetzen.
`^R`	Bildschirm neu zeichnen.
`s`	Zeichen unter dem Cursor durch die eingetippten Zeichen ersetzen.
`S`	Ganze Zeile ändern.
`^S`	Unbenutzt. (Auf manchen Terminals Anhalten der Datenübertragung.)
`t`	Cursor ein Zeichen vor dem nächsten Vorkommen des nächsten eingetippten Zeichens positionieren.
`T`	Cursor ein Zeichen nach dem letzten Vorkommen des nächsten eingetippten Zeichens positionieren.
`^T`	Unbenutzt im Befehlsmodus; im Einfügemodus Sprung auf die nächste Tabulatorposition.
`u`	Letzte Änderung rückgängig machen.
`U`	Aktuelle Zeile restaurieren.
`^U`	Bildschirm eine halbe Fenstergröße nach oben scrollen.
`v`	Unbenutzt.
`V`	Unbenutzt.
`^V`	Unbenutzt im Befehlsmodus; im Einfügemodus: nächstes Zeichen uninterpretiert einfügen.
`w`	Zum Anfang des nächsten Wortes springen.
`W`	Zum Anfang des nächsten Wortes springen, dabei keine Satzzeichen berücksichtigen.
`^W`	Unbenutzt im Befehlsmodus; im Einfügemodus Rücksprung auf den Anfang des aktuellen Wortes.
`x`	Zeichen unter dem Cursor löschen.
`X`	Zeichen vor dem Cursor löschen.
`^X`	Unbenutzt.
`y`	Kopier- oder Ausschneideoperator.
`Y`	Kopie der aktuellen Zeile anlegen.
`^Y`	Eine weitere Zeile am Bildschirmanfang anzeigen.

`z`	Aktuelle Zeile neu positionieren. Auf **z** muß eine dieser Tasten folgen: RETURN (Zeile auf den Bildschirmanfang positionieren), . (Zeile auf die Bildschirmmitte positionieren) oder - (Zeile auf das Bildschirmende positionieren).
`ZZ`	Änderungen abspeichern und Editor beenden.
`^Z`	**vi** unterbrechen.

vi-Einstellungen

Dieser Abschnitt behandelt die folgenden Punkte:

- Der :**set**-Befehl
- Optionen des :**set**-Befehls
- Beispielhafte *.exrc*-Datei

Der :set-Befehl

Der :**set**-Befehl ermöglicht es, Einstellungen vorzunehmen, die diverse Operationen des **vi** beeinflussen. Die Optionen können entweder in der *.exrc*-Datei eingetragen oder in einer **vi**-Sitzung eingestellt werden.

Der Doppelpunkt darf nicht eingegeben werden, wenn der Befehl in der *.exrc* steht.

`:set` *x*	Option *x* einschalten.
`:set no`*x*	Option *x* ausschalten.
`:set` *x*=*Wert*	Option *x* auf den Wert *Wert* setzen.
`:set`	Geänderte Optionen anzeigen.
`:set all`	Alle Optionen anzeigen.
`:set` *x*?	Wert der Option *x* anzeigen.

Optionen des :set-Befehls

In der folgenden Tabelle werden die Optionen des :**set**-Befehls beschrieben. Die erste Spalte enthält dabei neben dem Optionsnamen eine evtl. mögliche Abkürzung. In der zweiten Spalte finden Sie den Defaultwert und in der dritten Spalte eine Beschreibung des Verhaltens von **vi**, wenn die jeweilige Option eingeschaltet ist.

Option	Default	Beschreibung
autoindent (**ai**)	**noai**	Im Einfügemodus jede Zeile auf die gleiche Ebene wie die vorige oder folgende einrücken. Wird zusammen mit der **shiftwidth**-Option verwendet.
autoprint (**ap**)	**ap**	Anzeigen der Änderungen nach jedem Editor-Befehl (bei globalen Ersetzungen Anzeigen der letzten Ersetzung).
autowrite (**aw**)	**noaw**	Automatisches Speichern der Datei vor dem Öffnen einer anderen Datei mit :**n** oder der Ausführung eines Linux-Befehls mit :**!**, sofern die Datei verändert worden ist.

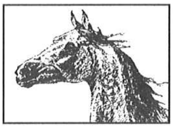
Option	Default	Beschreibung
beautify (bf)	nobf	Während der Eingabe alle Control-Zeichen außer Tabulator und Zeilensprung ignorieren.
directory= (dir)	/tmp	Name des Verzeichnisses, in dem **ex** temporäre Bufferdateien ablegt. Das Verzeichnis muß schreibbar sein.
edcompatible	noedcompatible	Verwendung **ed**-ähnlichen Verhaltens bei Substitutionsbefehlen.
errorbells (eb)	errorbells	Ausgeben einer akustischen Warnung, wenn ein Fehler aufgetreten ist.
exrc (ex)	noexrc	Ausführung von *.exrc*-Dateien außerhalb des Home-Verzeichnisses des Benutzers erlauben.
hardtabs= (ht)	8	Schrittweite für Hardware-Tabulatoren des Terminals einstellen.
ignorecase (ic)	noic	Groß-/Kleinschreibung während Suchoperationen nicht berücksichtigen.
lisp	nolisp	Einrückungen entsprechend den Lisp-Konventionen vornehmen. ( ), { }, [[und]] werden entsprechend den Lisp-Bedeutungen modifiziert.
list	nolist	Tabulatoren als ^I und Zeilenenden als $ ausgeben. (Mit **list** kann überprüft werden, ob es sich bei einem Zeichen um ein Tabulator- oder ein Leerzeichen handelt.)
magic	magic	Die Platzhalterzeichen **.** (Punkt), * (Stern) und [ ] (eckige Klammern) haben ihre besondere Bedeutung in Mustervergleichen.
mesg	mesg	Systemmeldungen werden während des Editierens mit **vi** angezeigt.
number (nu)	nonu	Anzeigen der Zeilennummern am linken Bildrand während des Editierens.
open	open	**ex**-Open-Modus darf verwendet werden.
optimize (opt)	noopt	Wagenrückläufe an Zeilenenden werden unterdrückt, wenn mehrere Zeilen ausgegeben werden. Dies beschleunigt die Ausgabe auf primitiven Terminals, wenn Zeilen mit führenden Leer- oder Tabulatorzeichen ausgegeben werden.
paragraphs= (para)	IPLPPPQPLI pplpipbp	Angeben der Absatzbegrenzungen beim Bewegen mit { und }. Die Zeichenpaare im angegebenen Wert entsprechen den Namen der nroff/troff-Makros, die Absätze einleiten.
prompt	prompt	Anzeigen der **ex**-Eingabeaufforderung beim Q-Befehl.

Option	Default	Beschreibung
readonly (ro)	noro	Jede Schreiboperation schlägt fehl, wenn sie nicht mit **!** kombiniert wird (im Zusammenhang mit **w**, **ZZ** und **autowrite**).
redraw (re)	noredraw	Das Terminal baut den Bildschirm nach jeder Editieroperation neu auf, d.h., im Einfügemodus werden bereits vorhandene Zeichen weitergeschoben und auf gelöschte Zeilen folgende Zeilen unmittelbar nachgezogen. Der Default hängt von der Verbindungsgeschwindigkeit und dem Terminaltyp ab. **noredraw** ist sinnvoll bei einer langsamen Verbindung und einem primitiven Terminal: Gelöschte Zeilen werden als @ angezeigt, und eingefügter Text überschreibt zunächst den vorhandenen Text, bis auf ESC gedrückt wird.
remap	remap	Erlauben von verschachtelten Abbildungen.
report=	5	Ausgeben einer Meldung in der letzten Zeile, wenn Editieroperationen durchgeführt werden, die sich über mehr als die angegebene Anzahl von Zeilen erstrecken. Beispielsweise wird bei **6dd** die Meldung „6 lines deleted" ausgegeben.
scroll=	<1/2 Fenster>	Textmenge, die automatisch gescrollt wird.
sections= (sect)	SHNHH HU	Definition der Abschnittsbegrenzungen für Bewegungen mit [[und]]. Die angegebenen Zeichenpaare entsprechen den nroff/troff-Makros, die Abschnitte einleiten.
shell= (sh)	/bin/sh	Pfadname der Shell, die bei Shell-Befehlen (**:!** und **:sh**) verwendet wird. Der Defaultwert ist der Wert der SHELL-Umgebungsvariablen.
shiftwidth= (sw)	8	Angeben der Anzahl von Leerzeichen, die bei Einrückebefehlen (**^T**, **^D**, **>>**, und **<<**) eingefügt werden sollen.
showmatch (sm)	nosm	Wenn eine) oder eine } eingegeben wird, springt der Cursor kurz auf die dazugehörige (oder { und wieder zurück. Wenn die passende Klammer nicht auf dem Bildschirm zu sehen ist, wird eine akustische Warnung ausgegeben. Sehr nützlich beim Programmieren.
showmode	noshowmode	Ausgeben einer Meldung in der letzten Bildschirmzeile im Einfügemodus über die Art des Einfügens (bspw. „Open Mode" oder „Append Mode").
slowopen (slow)		Die Anzeige während des Eingebens nicht aktualisieren. Der Default ist abhängig von der Verbindungsgeschwindigkeit und dem Terminaltyp.

Option	Default	Beschreibung
tabstop= (ts)	8	Angeben der Anzahl von Leerzeichen, die bei Tabulatorzeichen während des Editierens eingefügt werden. (Drucker benutzen davon unabhängig einen Wert von 8.)
taglength= (tl)	0	Angeben der Anzahl von Zeichen, die bei der Überprüfung von Tags maßgeblich sind. Der Default (0) besagt, daß alle Zeichen maßgeblich sind.
tags=	tags /usr/lib/tags	Angeben des Pfadnamens von Tag-Dateien (siehe Beschreibung des **ctags**-Befehls in Kapitel 12). Default ist die Datei **tags** im aktuellen Verzeichnis sowie */usr/lib/tags*.
term=		Einstellen des Terminaltyps.
terse	noterse	Ausgeben kürzerer Fehlermeldungen.
timeout (to)	timeout	„time out" für Tastenbindungen nach einer Sekunde.
ttytype=		Einstellen des Terminaltyps. Der Default ist der Wert der TERM-Umgebungsvariablen.
warn	warn	Ausgeben der Meldung „No write since last change".
window= (w)		Anzeigen einer bestimmten Anzahl von Zeilen auf dem Bildschirm. Der Default ist abhängig von der Verbindungsgeschwindigkeit und dem Terminaltyp.
wrapmargin= (wm)	0	Definieren des rechten Randes. Wenn dieser Wert größer als null ist, werden automatisch Wagenrückläufe als Zeilenumbrüche eingefügt.
wrapscan (ws)	ws	Es wird über Anfang bzw. Ende der Datei hinweg gesucht.
writeany (wa)	nowa	Es darf in jede Datei abgespeichert werden.

Beispielhafte .exrc-Datei

```
set nowrapscan wrapmargin=7
set sections=SeAhBhChDh nomesg
map q :w^M:n^M
map v dwElp
ab ORA O'Reilly Verlag.
```



KAPITEL 8

Der Editor ex

ex ist ein zeilenorientierter Editor, der die Basis für den bildschirmorientierten Editor **vi** bildet. **ex**-Befehle wirken entweder auf die aktuelle Zeile oder einen Bereich von Zeilen in einer Datei. Unter Linux wird **ex** oft auch **hex** genannt.

Sie werden **ex** meistens aus dem **vi** heraus benutzen. Im **vi** werden **ex**-Befehle mit einem Doppelpunkt eingeleitet und mit RETURN abgeschlossen.

ex kann aber auch eigenständig, also von der Kommandozeile, aufgerufen werden — genau wie **vi** auch. Außerdem können Sie den Q-Befehl im **vi** verwenden, um **vi** zu verlassen und zum **ex** zu wechseln.

In diesem Kapitel werden die folgenden Punkte behandelt:

- Syntax der **ex**-Befehle
- Alphabetisch sortierte Zusammenfassung der Befehle

Weitere Informationen finden Sie in *Textbearbeitung mit dem vi-Editor* von Linda Lamb.

Syntax der ex-Befehle

Ein **ex**-Befehl wird vom **vi** in folgender Form eingegeben:

:[*Adresse*] *Befehl* [*Optionen*]

Am einleitenden : erkennt **vi**, daß es sich um einen **ex**-Befehl handelt. Während Sie den Befehl eingeben, wird er in der Statuszeile angezeigt. Sie schließen die Befehlseingabe durch Drücken der RETURN-Taste ab. *Adresse* bezeichnet die Zeile oder die Zeilen, auf der oder denen *Befehl* arbeiten soll. Was *Optionen* und *Adresse* genau bedeuten, lesen Sie im folgenden; die **ex**-Befehle werden in der alphabetischen Übersicht beschrieben.

Sie können **ex** folgendermaßen beenden:

:x	Änderungen speichern und beenden.
:q!	Beenden, ohne zu speichern.
:vi	**vi** aufrufen.

Optionen

!	Bezeichnet eine Variante des Befehls, die vom normalen Verhalten abweicht.
Anzahl	Gibt an, wie oft der Befehl wiederholt werden soll. Im Gegensatz zu **vi**-Befehlen kann *Anzahl* den ex-Befehlen nicht vorangestellt werden, da **ex** eine vorangestellte Zahl als Zeilenangabe interpretiert. So löscht beispielsweise **d3** drei Zeilen einschließlich der aktuellen, **3d** aber die Zeile Nummer 3.
Datei	Der Name der Datei, die durch den Befehl bearbeitet werden soll. % bezeichnet die aktuelle Datei, # die zuletzt bearbeitete.

Adressen

Wenn keine Adresse angegeben wird, wirkt der Befehl auf die aktuelle Zeile. Wenn ein Zeilenbereich bearbeitet werden soll, muß folgendes Format verwendet werden:

x,*y*

wobei *x* und *y* die jeweils erste und letzte Zeile des betroffenen Bereichs darstellen (*x* muß kleiner oder gleich *y* sein). *x* und *y* können Zeilennummern oder symbolische Ausdrücke sein. Wenn ; anstelle von , verwendet wird, wird die aktuelle Zeile auf *x* gesetzt, bevor *y* ausgewertet wird. Die Schreibweise **1,$** bezeichnet genau wie % alle Zeilen der Datei.

Adreßsymbole

`1,$`	Alle Zeilen der Datei
`%`	Alle Zeilen, gleich **1,$**
`x,y`	Die Zeilen von *`x`* bis *`y`*
`x;y`	Die Zeilen von *`x`* bis *`y`*, wobei die aktuelle Zeile auf *`x`* gesetzt wird
`0`	Dateianfang
`.`	Aktuelle Zeile
`n`	Absolute Zeilennummer *`n`*
`$`	Letzte Zeile
`x-n`	*`n`* Zeilen vor *`x`*
`x+n`	*`n`* Zeilen nach *`x`*
-[*`n`*]	Eine oder *`n`* Zeilen weiter vorn
+[*`n`*]	Eine oder *`n`* Zeilen weiter hinten
'*`x`*	Mit der Markierung *`x`* versehene Zeile
`''`	Vorherige Markierung
/*`Suchmuster`*/	Vorwärts bis zur nächsten Zeile, auf die *`Suchmuster`* paßt
?*`Suchmuster`*?	Zurück bis zur nächsten Zeile, auf die *`Suchmuster`* paßt

Siehe Kapitel 5, um weitere Informationen zu Mustervergleichen zu erhalten.

Alphabetisch sortierte Übersicht der ex-Befehle

ex-Befehle können so weit verkürzt werden, wie der Name noch eindeutig ist. In der folgenden Übersicht steht der volle Name des Befehls am Rand und die kürzestmögliche Abkürzung in der syntaktischen Beschreibung. Die Beispiele sind so dargestellt, wie sie aus dem **vi** heraus eingegeben werden würden, also mit Doppelpunkt.

---

ab [*String Text*] — **abbrev**

Definiert *String* als Abkürzung für *Text*, d.h., wenn *String* eingegeben wird, wird dieser durch *Text* ersetzt. Wenn keine Parameter angegeben werden, werden alle aktuellen Abkürzungen angezeigt.

Beispiele

Anmerkung: ^M erscheint bei Eingabe von **CTRL-V**, gefolgt von RETURN.

```
:ab ora O'Reilly Verlag.
:ab id Name:^MPosition:^MTelefon:
```

[*Adresse*] a[!] — **append**
Text
.

Anhängen von *Text* an der *Adresse* oder an der aktuellen Adresse, wenn keine angegeben ist. Durch Hinzufügen von ! wird die **autoindent**-Einstellung, die während des Einfügens verwendet wird, umgeschaltet, d.h., wenn **autoindent** vorher eingeschaltet war, ist es während der Ausführung dieses Befehls abgeschaltet und umgekehrt.

---

ar — **args**

Ausgeben der Dateinamen-Argumente (die Liste der zu editierenden Dateien). Die aktuelle Datei wird in eckigen Klammern angezeigt ([]).

[*Adresse*] c[!] — **change**
Text
.

Ersetzen der angegebenen Zeilen durch *Text*. Durch Anhängen eines ! wird die **autoindent**-Einstellung während des Ersetzens umgeschaltet.

[*Adresse*] co *Ziel* — **copy**

Kopieren der Zeilen in *Adresse* nach *Ziel*. Dies ist identisch mit dem t-Befehl.

Beispiel

`:1,10 co 50` *Kopieren der ersten zehn Zeilen hinter Zeile 50*

delete	[*Adresse*] d [*Buffer*] Löschen der Zeilen in *Adresse*. Wenn *Buffer* angegeben wird, werden die Zeilen dort abgespeichert oder angehängt. ***Beispiele*** `:/Teil I/,/Teil II/-1d` *Bis zur Zeile oberhalb von „Teil II" löschen* `:/main/+d` *Zeile unterhalb von „main" löschen* `:.,$d` *Von der aktuellen Zeile bis zum Ende löschen*
edit	e[!] [+*n*] [*Datei*] Neue Datei editieren. Wenn ein ! angehängt wird, werden alle Änderungen an der aktuellen Datei verworfen. Wenn keine Datei angegeben wird, wird eine weitere Kopie der aktuellen Datei editiert. Wird das +*n*-Argument angegeben, wird in der neuen Datei auf Zeile *n* gesprungen. ***Beispiele*** `:e Datei` `:e#` *Rücksprung zur vorher bearbeiteten Datei* `:e!` *Alle Änderungen seit dem letzten Speichern verwerfen*
file	f [*Dateiname*] Ändern des Namens der aktuellen Datei in *Dateiname*. Die Datei gilt als nicht bearbeitet. Wenn *Dateiname* nicht angegeben wird, wird der aktuelle Zustand der Datei ausgegeben. ***Beispiel*** `:f %.neu`
global	[*Adresse*] g[!]/*Suchmuster*/[*Befehle*] Ausführen der *Befehle* auf allen Zeilen, die *Suchmuster* enthalten, oder, wenn *Adresse* angegeben wird, auf allen Zeilen dieses Bereichs. Wenn *Befehle* weggelassen wird, werden alle entsprechenden Zeilen ausgegeben. Mit ! werden die Befehle auf allen Zeilen ausgeführt, die das Suchmuster *Suchmuster* nicht enthalten. ***Beispiele*** `:g/Linux/p` *Ausgeben aller Zeilen, die „Linux" enthalten* `:g/Name:/s/tom/Tom/` *In allen Zeilen, die „Name:" enthalten, „tom" durch „Tom" ersetzen*
insert	*Adresse* i[!] *Text* . Text an der Zeile vor der angegebenen Adresse einfügen. Wenn *Adresse* nicht angegeben wird, wird der Text an der aktuellen Adresse eingefügt. Mit ! wird die Einstellung von **autoindent** für das Einfügen umgeschaltet.

[*Adresse*] j[!] [*Anzahl*] — join

Der Text in *Adresse* wird auf einer Zeile zusammengefaßt, wobei hinter jedem Punkt (.) zwei Leerzeichen eingefügt werden, hinter) keine Leerzeichen stehen und ansonsten alle Leerzeichenfolgen zu einer zusammengefaßt werden. Mit ! werden die Leerzeichen nicht verändert.

Beispiel

`:1,5j!`	*Die ersten fünf Zeilen in einer zusammenfassen, Leerzeichen nicht verändern*

[*Adresse*] k *Zeichen* — k

Adresse mit *Zeichen* markieren. Zu dieser Zeile kann später zurückgesprungen werden.

[*Adresse*] l [*Anzahl*] — list

Ausgeben der angegebenen Zeilen, so daß Tabulatorzeichen als ^I und Zeilenenden als $ erscheinen. l ist eine temporäre Version von **:set list**.

map[!] [*Zeichen Befehle*] — map

Definition eines Tastaturmakros namens *Zeichen* als Befehlsfolge *Befehle*. *Zeichen* ist normalerweise ein einziges Zeichen oder die Zeichenfolge *#n*, die Funktionstasten bezeichnet. Mit ! wird ein Makro für den Einfügemodus definiert. Wenn kein Argument angegeben wird, werden die aktuell definierten Makros angezeigt.

Beispiele

`:map K dwwP`	*Vertauschen zweier Wörter*
`:map q :w^M:n^M`	*Aktuelle Datei abspeichern und zur nächsten wechseln*
`:map! + ^[bi(^[ea)`	*Vorhergehendes Wort in runde Klammern einschließen*

[*Adresse*] ma *Zeichen* — mark

Adresse mit *Zeichen* markieren. Zu dieser Zeile kann später zurückgesprungen werden. Wie **k**.

[*Adresse*] m *Ziel* — move

Verschieben der mit *Adresse* bezeichneten Zeilen nach *Ziel*.

`:.,/Note/m /ENDE/`	*Textblock hinter die Zeile mit „ENDE“ verschieben*

n[!] [[+*Befehl*] *Dateiliste*] — next

Bearbeiten der nächsten auf der Kommandozeile eingegebenen Datei. Mit **args** kann die Dateiliste angezeigt werden. Wenn *Dateiliste* angegeben wird, wird die aktuelle Dateiliste durch *Dateiliste* ersetzt und mit dem Editieren der

→

next ←	ersten Datei dieser Liste begonnen. Wenn *Befehl* angegeben wird (darf keine Leerzeichen enthalten), wird dieser Befehl nach dem Editieren der ersten Datei ausgeführt. ***Beispiel*** `:n kap*` *Editieren aller „kapitel"-Dateien*
number	[*Adresse*] **nu** [*Anzahl*] Ausgeben jeder Zeile in *Adresse* mit der vorangestellten Zeilennummer. Anstelle von **number** kann dieser Befehl auch mit # aufgerufen werden. *Anzahl* gibt an, wie viele Zeilen angezeigt werden sollen.
open	[*Adresse*] **o** [/*Suchmuster*/] Starten des Open-Modus von **vi** an den mit *Adresse* angegebenen Zeilen oder den Zeilen, auf die *Suchmuster* paßt. Der Open-Modus kann mit **Q** aufgerufen und wieder verlassen werden. Im Open-Modus können die normalen **vi**-Befehle verwendet werden, wirken aber nur auf jeweils eine Zeile. Dies kann bei langsamen Terminalverbindungen nützlich sein.
preserve	**pre** Speichern des aktuellen Buffers, als ob das System abgestürzt wäre.
print	[*Adresse*] **p** [*Anzahl*] Ausgeben der Zeilen, die durch *Adresse* spezifiziert werden. *Anzahl* gibt an, wie viele Zeilen ausgegeben werden sollen. Eine andere Abkürzung für diesen Befehl ist **P**. ***Beispiel*** `:100;+5p` *Anzeigen von Zeile 100 und den folgenden fünf Zeilen*
put	[*Adresse*] **pu** [*Zeichen*] Zurückholen der Zeilen, die zuletzt aus dem Buffer *Zeichen* gelöscht oder kopiert worden sind. Wenn *Zeichen* nicht angegeben wird, werden die zuletzt gelöschten oder kopierten Zeilen zurückgeholt.
quit	**q**[**!**] Beenden der aktuellen Editier-Sitzung. Mit **!** können sämtliche Änderungen seit dem letzten Speichern verworfen werden. Wenn noch Dateien zur Bearbeitung ausstehen, auf die bisher noch nicht zugegriffen wurde, muß die Sitzung durch Eingabe von **q!** oder zweimaliges Eingeben on **q** beendet werden.

[*Adresse*] r *Datei* Einfügen von *Datei* hinter *Adresse*. Wenn *Datei* nicht angegeben wird, wird der aktuelle Dateiname verwendet. ***Beispiel*** `:0r $HOME/data` *Einfügen einer Datei am Anfang der aktuellen Datei*	read
[*Adresse*] r !*Befehl* Einfügen der Ausgabe des Linux-*Befehls* an der Zeile hinter *Adresse*. ***Beispiel*** `:$r !cal` *Einfügen eines Kalenders am Ende der aktuellen Datei*	read
rec [*Datei*] *Datei* aus dem Sicherungsbereich des Systems retten.	recover
rew[!] Zurücksetzen der Dateiliste und Editieren der ersten Datei auf dieser Liste. Mit ! werden alle seit dem letzten Speichern gemachten Änderungen verworfen.	rewind

se *Parameter1 Parameter2 ...*
 	
Jeder Parameter setzt einen Wert für eine Option. Wenn kein Parameter angegeben wird, werden die Werte aller Optionen, die nicht die Defaultwerte haben, ausgegeben. Bei zweiwertigen Optionen können die Parameter als *option* oder no*option*, andere Optionen können mit der Schreibweise *option*=*Wert* zugewiesen werden. Mit all werden die aktuellen Einstellungen angezeigt.
 	
Beispiele
 	
`:set nows wm=10`
`:set all`
	set

sh Erzeugen einer neuen Shell. Wenn diese Shell beendet wird, wird das Editieren fortgesetzt.	shell
so *Datei* Einlesen und Ausführen der Befehle in *Datei*. ***Beispiel*** `:so $HOME/.exrc`	source

substitute	[*Adresse*] s [/*Suchmuster*/*Ersetzungsmuster*/] [*Optionen*] [*Anzahl*] Ersetzen jedes Vorkommens von *Suchmuster* in den angegebenen Zeilen durch *Ersetzungsmuster*. Wenn *Suchmuster* und *Ersetzungsmuster* weggelassen werden, wird die letzte Ersetzung wiederholt. *Anzahl* gibt die Anzahl der Zeilen ab *Adresse* an, auf denen ersetzt werden soll. Weitere Beispiele finden Sie in „Beispiele zum Suchen und Ersetzen" in Kapitel 5. ***Optionen*** c — Vor jeder Ersetzung nachfragen. g — Jedes Vorkommen (und nicht nur das erste) von *Suchmuster* auf jeder Zeile ersetzen. p — Ausgeben der letzten Zeile, auf der eine Ersetzung vorgenommen wurde. ***Beispiele*** `:1,10s/ja/nein/g` — *In den ersten zehn Zeilen ersetzen* `:%s/[Hh]allo/Tach/gc` — *Globale Ersetzungen bestätigen* `:s/Prolog/\U&/ 3` — *In den folgenden drei Zeilen das erste Vorkommen von „Prolog" in Großbuchstaben umwandeln.*
t	[*Adresse*] t *Ziel* Kopieren der in *Adresse* angegebenen Zeilen nach *Ziel*. Identisch mit **copy**. ***Beispiel*** `:%t$` — *Datei kopieren und am Ende anhängen*
tag	[*Adresse*] ta *Tag* Datei, die *Tag* enthält, bearbeiten. ***Beispiel*** ctags aufrufen und zur Datei wechseln, die *meine_funktion* enthält: `:!ctags *.c` `:tag` *`meine_funktion`*
unabbreviate	una *Wort* Aufheben der Abkürzung für *Wort*.
undo	u Aufheben der beim letzten Editierbefehl gemachten Änderungen.
unmap	unm[!] *Zeichen* *Zeichen* von der Liste der Tastaturmakros entfernen. Mit ! wird ein Tastaturmakro für den Eingabemodus entfernt.

[*Adresse*] v/*Suchmuster*/[*Befehle*] Ausführen der *Befehle* auf allen Zeilen, die *Suchmuster nicht* enthalten. Wenn *Befehle* angegeben wird, werden alle passenden Zeilen angezeigt. **v** entspricht **g!**. ***Beispiel*** `:v/#include/d` *Alle Zeilen außer „#include"-Anweisungen löschen*	v
ve Ausgeben der Versionsnummer des Editors.	**version**
[*Adresse*] **vi** [*Typ*] [*Anzahl*] Aufrufen des Visual-Modus (**vi**) an der Zeile *Adresse*. Der Modus wird mit Q wieder beendet. *Typ* kann entweder -, ^ oder . sein (siehe dazu den z-Befehl). *Anzahl* gibt die anfängliche Fenstergröße an.	**visual**
vi [+*n*] Datei *Datei* im Visual-Modus (**vi**) editieren (optional ab Zeile *n*).	**vi**
[*Adresse*] **w**[!] [[>>] *Datei*] Schreiben der mit *Adresse* bezeichneten Zeilen in *Datei*. Wenn *Adresse* nicht angegeben wird, wird der gesamte Buffer geschrieben. Wenn auch *Datei* weggelassen wird, wird der Bufferinhalt in die aktuelle Datei gespeichert. Wenn >> *Datei* verwendet wird, wird der Inhalt an das Ende der existierenden *Datei* angehängt. Mit ! wird jeder bereits vorhandene Inhalt von *Datei* überschrieben.	**write**
[*Adresse*] **w** !*Befehl* Die durch *Adresse* bezeichneten Zeilen an den *Befehl* schicken. ***Beispiele*** `:1,10w namen_liste` *Die ersten zehn Zeilen an den Befehl* `namen_liste` *schicken* `:50w >> namen_liste` *Jetzt Zeile 50 anhängen*	**write**
wq[!] Datei abspeichern und beenden in einem. Mit ! wird ein eventuell vorhandener Inhalt der Datei ohne Rückfrage überschrieben.	**wq**
x Abspeichern der Datei, wenn sie seit dem letzten Speichern verändert wurde; danach Beenden.	**xit**

yank

[*Adresse*] ya [*Zeichen*] [*Anzahl*]

Kopieren der durch *Adresse* bezeichneten Zeilen in den benannten Buffer *Zeichen*. Wenn *Zeichen* nicht angegeben wird, werden die Zeilen in die allgemeine Zwischenablage kopiert. *Anzahl* gibt, an wie viele Zeilen beginnend mit *Adresse* kopiert werden sollen.

Beispiel

```
:101,200 ya a
```

---

z

[*Adresse*] z [*Typ*] [*Anzahl*]

Ausgeben eines Textfensters. Die durch *Adresse* angegebene Zeile erscheint am oberen Rand. *Anzahl* gibt an, wie viele Zeilen angezeigt werden sollen.

Typ

+	Angegebene Zeile am oberen Fensterrand anzeigen (Default).
-	Angegebene Zeile am unteren Fensterrand anzeigen.
.	Angegebene Zeile in der Fenstermitte anzeigen.
^	Ein Fenster nach oben bewegen.
=	Angegebene Zeile in der Fenstermitte anzeigen und zur aktuellen Zeile machen.

!

[*Adresse*] !*Befehl*

Ausführen des Linux-*Befehls* in einer Shell. Wenn *Adresse* angegeben wird, werden die damit bezeichneten Zeilen als Standardeingabe an *Befehl* geschickt und durch die Ausgabe des Befehls ersetzt.

Beispiele

`:!ls`	*Anzeigen der Dateien im aktuellen Verzeichnis*
`:11,20!sort -f`	*Zeilen 11–20 der aktuellen Datei sortieren*

=

[*Adresse*] =

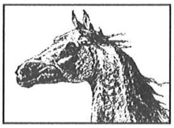
Ausgeben der Zeilennummer der nächsten Zeile, auf die *Adresse* paßt. Wenn *Adresse* nicht angegeben wird, wird die Nummer der letzten Zeile ausgegeben.

<>

[*Adresse*] < [*Anzahl*]
[*Adresse*] > [*Anzahl*]

Verschieben der durch *Adresse* angegebenen Zeilen nach links (<) oder rechts (>). Bei einer Verschiebung nach links werden nur Leer- und Tabulatorzeichen entfernt. *Anzahl* gibt an, wie viele Zeilen beginnend mit *Adresse* verschoben werden sollen.

Adresse Ausgeben der mit *Adresse* bezeichneten Zeile.	*Adresse*
RETURN Ausgeben der nächsten Zeile der Datei.	RETURN
& [*Optionen*] [*Anzahl*] Wiederholen des letzten Ersetzungsbefehls (`s`). *Anzahl* gibt an, in wie vielen Zeilen beginnend mit *Adresse* ersetzt werden soll. ***Beispiele*** `:s/Gemahnt/Bezahlt/` *Auf der aktuellen Zeile ersetzen* `:g/Status/&` *Ersetzung auf allen „Status"-Zeilen wiederholen*	&
[*Adresse*] ~ [*Anzahl*] Ersetzen des letzten regulären Ausdrucks durch das letzte Ersetzungsmuster aus einem Ersetzungsbefehl.	~



KAPITEL 9

Der Editor sed

In diesem Kapitel finden Sie die folgenden Abschnitte:

- Überblick über **sed**
- Kommandozeilensyntax
- Syntax der **sed**-Befehle
- Nach Gruppen sortierte Zusammenfassung der **sed**-Befehle
- Alphabetisch sortierte Zusammenfassung der **sed**-Befehle

Mehr Informationen zu **sed** finden Sie in *sed & awk*, zweite Auflage, von Dale Dougherty und Arnold Robbins.

Überblick über sed

sed ist ein nicht-interaktiver oder datenflußorientierter Editor.* **sed** interpretiert ein sogenanntes Skript und führt die darin angegebenen Aktionen aus. **sed** ist datenflußorientiert wie viele andere UNIX-Befehle, weil die Eingabe gewissermaßen durch das Programm hindurch zur Standardausgabe fließt. Ein anderes Beispiel für einen datenflußorientierten Befehl ist **sort**, ein Gegenbeispiel **vi**. Die Eingaben für **sed** kommen normalerweise aus einer Datei, können aber auch von der Tastatur entgegengenommen werden. Die Ausgabe erfolgt standardmäßig über den Bildschirm, kann aber auch in eine Datei umgelenkt werden.

Gängige Anwendungsbeispiele für **sed** sind:

- Automatisches Editieren einer oder mehrerer Dateien
- Sich wiederholende Editiervorgänge auf vielen Dateien vereinfachen
- Konvertierungsprogramme schreiben

*Der Name **sed** ist eine Abkürzung des englischen „stream editor".

sed arbeitet folgendermaßen:

- Jede Eingabezeile wird in einen gesonderten Speicherbereich kopiert.
- Alle Editierbefehle eines **sed**-Skriptes werden auf diesen Speicherbereich angewendet.
- Editierbefehle werden auf alle Zeilen angewendet, es sei denn, sie sind durch Angabe einer Zeilenadresse auf einen bestimmten Bereich beschränkt.
- Wenn ein Befehl die Eingabezeile verändert, werden die folgenden Befehle auf die geänderte und nicht auf die originale Zeile angewendet.
- Die ursprüngliche Eingabezeile bleibt unverändert, weil die Editierbefehle eine Kopie der Eingabezeile verändern. Diese Kopie wird auf die Standardausgabe geschrieben, kann aber auch in eine Datei umgeleitet werden.

Kommandozeilensyntax

sed kann auf zweierlei Weisen aufgerufen werden:

> **sed** [*Optionen*] '*Befehl*' *Datei(en)*
> **sed** [*Optionen*] -f *Skriptdatei Datei(en)*

Mit der ersten Schreibweise kann ein Editierbefehl auf der Kommandozeile in einfachen Anführungsstrichen angegeben werden. Mit der zweiten Form kann eine Skriptdatei angegeben werden, der **sed** die auszuführenden Befehle entnimmt. Wenn keine zu bearbeitenden Dateien angegeben werden, liest **sed** von der Standardeingabe.

sed kennt die folgenden Optionen:

-e *Befehl*
: Das nächste Argument ist ein Editierbefehl. Diese Option wird nur benötigt, wenn zwei oder mehr Editierbefehle auf der Kommandozeile angegeben werden sollen.

-f *Datei*
: Das nächste Argument ist eine Skriptdatei mit Editierbefehlen.

-g Alle Substitutionen sind global.

-n Unterdrücken der defaultmäßigen Ausgabe; **sed** gibt nur die Zeilen aus, die mit dem **p**-Befehl oder der **p**-Option des **s**-Befehls angegeben wurden.

Syntax der sed-*Befehle*

sed-Befehle haben die Form:

> [*Adresse*][,*Adresse*][!]*Befehl* [*Argumente*]

sed-Befehle bestehen aus Adressen und Befehlen. Befehle bestehen aus einem einzelnen Buchstaben oder Sonderzeichen und werden später alphabetisch und nach Gruppen sortiert beschrieben. Einzelne Befehle können Argumente haben: eine Markierung beim **b**- und **t**-Befehl, ein Dateiname beim **r**- und **w**-Befehl und Optionen beim **s**-Befehl. Das Format der Adressen wird im folgenden Abschnitt beschrieben.

Musteradressierung

Bei jedem **sed**-Befehl können keine, eine oder zwei Adressen angegeben werden. Eine Adresse kann eine Zeilennummer, das Symbol **$** für die letzte Zeile oder ein regulärer Ausdruck, der in Schrägstrichen eingeschlossen wird (*/Suchmuster/*), sein. Reguläre Ausdrücke werden in Kapitel 5, *Mustervergleiche* beschrieben. Darüber hinaus kann **\n** angegeben werden, was auf jeden Zeilenumbruch im Arbeitsbereich von **sed** paßt (solche Zeilenumbrüche entstammen dem N-Befehl), nicht aber auf den Zeilenumbruch am Ende des Arbeitsbereiches.

Wenn beim Befehl angegeben wird:	***Wird der Befehl angewendet auf:***
Keine Adresse	Jede Eingabezeile.
Eine Adresse	Jede Zeile, die auf die Adresse paßt. Einige Befehle (**a**, **i**, **r**, **q** und **=**) können nur eine Adresse haben.
Zwei durch Kommata getrennte Adressen	Die erste passende Zeile und alle weiteren bis zur ersten Zeile, die auf die zweite Adresse paßt.
Eine Adresse, gefolgt von !	Alle Zeilen, die nicht auf diese Adresse passen.

Beispiele

`s/xx/yy/g`	*Alle Vorkommen auf allen Zeilen ersetzen*
`/BSD/d`	*Alle Zeilen löschen, die* `BSD` *enthalten*
`/^BEGIN/,/^END/p`	*Text zwischen* `BEGIN` *und* `END` *(einschließlich) ausgeben*
`/SAVE/!d`	*Alle Zeilen löschen, die nicht* `SAVE` *enthalten*
`/BEGIN/,/END/!s/xx/yy/g`	*Auf allen Zeilen ersetzen, die nicht im Bereich zwischen* `BEGIN` *und* `END` *liegen*

Geschweifte Klammern ({}) werden in **sed** verwendet, um Adressen ineinander zu verschachteln oder mehrere Befehle auf ein und dieselbe Adresse anzuwenden:

[*/Adresse/*][,*/Adresse/*]{
Befehl1
Befehl2
}

Die öffnende geschweifte Klammer muß am Zeilenende, die schließende allein auf einer Zeile stehen. Beachten Sie, daß den Klammern keine Leerzeichen folgen dürfen.

Nach Gruppen sortierte Zusammenfassung der sed-Befehle

In den folgenden Listen sind die **sed**-Funktionen nach Funktionsgruppen sortiert und knapp beschrieben. Vollständige Beschreibungen mit Syntax und Beispielen finden Sie in der folgenden alphabetisch sortierten Übersicht.

Elementare Editierfunktionen

a	Text am Ende einer Zeile anhängen.
c	Text ersetzen (normalerweise ein Textblock).
i	Text am Anfang einer Zeile einfügen.
d	Zeilen löschen.
s	Ersetzungen vornehmen.
y	Zeichen konvertieren (wie **tr** in Kapitel 1, *Linux-Benutzerbefehle*).

Zeileninformation

`=`	Anzeigen der Zeilennummer.
`l`	Anzeigen von Steuerzeichen in ASCII.
`p`	Anzeigen der Zeile.

Ein-/Ausgabebefehle

`n`	Aktuelle Zeile überspringen und mit der nächsten weitermachen.
`r`	Einlesen einer anderen Datei.
`w`	Schreiben in eine andere Datei.
`q`	Beenden des **sed**-Skriptes (keine weiteren Ausgaben).

Kopieren und Einfügen

`h`	Kopieren des Arbeitsbereichs in die Zwischenablage, deren vorheriger Inhalt verlorengeht.
`H`	Kopieren des Arbeitsbereichs in die Zwischenablage, dabei an dort vorhandene Daten anhängen.
`g`	Zurückholen der Zwischenablage in den Arbeitsbereich, dessen vorheriger Inhalt verlorengeht.
`G`	Zurückholen der Zwischenablage in den Arbeitsbereich, dabei an dort vorhandene Daten anhängen.
`x`	Inhalte von Arbeitsbereich und Zwischenablage austauschen.

Verzweigungsbefehle

`b`	Verzweigen nach *Sprungziel* oder zum Ende des Skriptes.
`t`	Wie **b**, aber nur nach einer Ersetzung verzweigen.
`:`*`Sprungziel`*	Sprungziel (*Sprungziel*) für **b** und **t**.

Verarbeiten mehrerer Eingabezeilen auf einmal

`N`	Einlesen einer weiteren Eingabezeile (dabei wird ein eingebetteter Zeilenwechsel eingefügt).
`D`	Alles bis zum eingebetteten Zeilenwechsel löschen.
`P`	Alles bis zum eingebetteten Zeilenwechsel ausgeben.

Alphabetisch sortierte Übersicht der sed-Befehle

:	:*Sprungziel* Benennen einer Zeile in einem Skript für Sprünge mit **b** oder **t**. *Sprungziel* darf bis zu sieben Zeichen enthalten.

[*/Suchmuster/*]= =

Ausgeben der Zeilennummer jeder Zeile, die *Suchmuster* enthält.

[*Adresse*]a\ a
Text

An jede Zeile, die durch *Adresse* beschrieben wird, wird *Text* angehängt. Wenn *Text* sich über mehr als eine Zeile erstreckt, müssen enthaltene Zeilensprünge durch einen Backslash versteckt werden. *Text* gilt als beendet durch den ersten nicht versteckten Zeilenwechsel. Der angehängte Text steht nicht im Arbeitsbereich zur Verfügung und kann somit auch nicht von nachfolgenden Befehlen verwendet werden. Das Ergebnis dieses Befehls wird nach Abschluß des Editiervorganges auf die Standardausgabe geschrieben, unabhängig davon, wie die aktuelle Zeile im Arbeitsbereich verändert wird.

Beispiel

```
$a\
Dies wird in die letzte Zeile der Datei geschrieben\
(angegeben durch $). Saemtliche Zeilenwechsel am Ende einer\
Zeile mit Ausnahme des letzten sind versteckt.
```

[*Adresse1*][,*Adresse2*]b[*Sprungziel*] b

Bedingungsloser Sprung nach *:Sprungziel* an einer anderen Stelle im Skript, d.h. der auf *Sprungziel* folgende Befehl ist der nächste Befehl, der auf die aktuelle Zeile angewendet wird. Wenn *Sprungziel* nicht angegeben wird, wird die Verarbeitung abgebrochen.

Beispiel

Zeilen zwischen .TS und .TE ignorieren; weitere Abarbeitung des Skripts nach .TE:

```
/^\.TS/,/^\.TE/b
```

[*Adresse1*][,*Adresse2*]c\ c
Text

Ersetzen der durch die Adresse angegebenen Zeilen durch *Text*. Wenn ein Bereich von Zeilen angegeben wird, werden alle Zeilen zusammen durch eine einzige Kopie von *Text* ersetzt. Zeilensprünge innerhalb von *Text* müssen mit Ausnahme des letzten durch eine Backslash versteckt werden. Der Inhalt des Arbeitsbereichs wird gelöscht und steht nachfolgenden Editierbefehlen nicht mehr zur Verfügung.

Beispiel

Ersetzen der ersten hundert Zeilen einer Datei:

```
1,100c\
\
<First 100 names to be supplied>
```

d [*Adresse1*][,*Adresse2*]**d**

Löschen der adressierten Zeilen aus dem Arbeitsbereich. Die entsprechenden Zeilen werden also nicht an die Standardausgabe weitergereicht. Die nächste Zeile wird eingelesen und mit dem ersten Befehl des Skripts bearbeitet.

Beispiel

Alle leeren Zeilen löschen:

```
/^$/d
```

D [*Adresse1*][,*Adresse2*]**D**

Löschen des ersten Teils (alles bis zu einem enthaltenen Zeilenwechsel) eines mehrzeiligen Arbeitsbereichs, der durch den **N**-Befehl angelegt wurde. Der Editiervorgang wird danach mit dem ersten Befehl des Skripts fortgesetzt. Wenn die Ausführung dieses Befehls zur Leerung des Arbeitsbereiches führt, wird eine neue Zeile eingelesen, als ob **d** ausgeführt worden wäre.

Beispiel

Mehrere leere Zeilen zu einer zusammenfassen:

```
/^$/{
N
/^\n$/D
}
```

g [*Adresse1*][,*Adresse2*]**g**

Einfügen des Inhalts der Zwischenablage (siehe **h**- und **H**-Befehle) in den Arbeitsbereich. Alles, was vorher im Arbeitsbereich enthalten war, wird gelöscht. Im Beispiel ist eine einfache Methode angegeben, Zeilen zu kopieren.

Beispiel

Das folgende Skript nimmt alle Zeilen, die das Wort *Position:* enthalten, und kopiert sie an eine durch einen Platzhalter markierte Stelle weiter hinten in der Datei. Der Platzhalter wird überschrieben.

```
/Position:/H
/<Diese Zeile wird mit der Positionsliste überschrieben>/g
```

G [*Adresse1*][,*Adresse2*]**G**

Wie **g**, aber der Text wird unterhalb der angegebenen Adresse eingefügt, anstatt den dortigen Text zu überschreiben. Das Beispiel zeigt eine einfache Methode, Text zu kopieren und wieder einzufügen.

Beispiel

Das folgende Skript nimmt alle Zeilen, die das Wort *Position:* enthalten, und verschiebt sie hinter eine durch einen Platzhalter markierte Stelle weiter hinten in der Datei. Die ursprünglichen *Position:*-Zeilen werden gelöscht.

G

```
/Position:/{
H
d
}
/Liste der Position:/G
```

h

[*Adresse1*][,*Adresse2*]h

Kopieren des Arbeitsbereiches in die Zwischenablage, einen speziellen temporären Buffer. Der vorherige Inhalt der Zwischenablage wird verworfen. **h** kann verwendet werden, um eine Zeile abzuspeichern, bevor sie editiert wird.

Beispiel

```
# Eine Zeile editieren, Veraenderungen ausgeben,
# Original noch einmal anzeigen
/Linux/{
h
s/.* Linux \(.*\) .*/\1:/
p
x
}
```

Beispiel-Eingabe:

```
Beschreibung des Linux ls-Befehls.
Beschreibung des Linux cp-Befehls.
```

Beispiel-Ausgabe:

```
ls:
Beschreibung des Linux ls-Befehls.
cp:
Beschreibung des Linux cp-Befehls.
```

H

[*Adresse1*][,*Adresse2*]H

Anhängen des Inhalts des Arbeitsbereiches sowie eines führenden Zeilenwechsels an den Inhalt der Zwischenablage. Auch wenn die Zwischenablage leer ist, wird der führende Zeilenwechsel mit eingefügt. **H** entspricht einem inkrementellen Kopiervorgang. Siehe dazu die Beispiele unter **g** und **G**.

i

[*Adresse1*]i\
Text

Einfügen von Text vor jeder Zeile, die durch *Adresse* bestimmt wird. (Siehe die Beschreibung zum **a**-Befehl, um nähere Informationen zum Format von *Text* zu erhalten.)

Beispiel

```
/Position 1/i\
Die fünf Positionen sind unten aufgeführt:
```

l	[*Adresse1*][,*Adresse2*]l Anzeigen des Inhalts des Arbeitsbereiches, wobei nicht druckbare Zeichen als ASCII-Zeichen dargestellt werden. Zu lange Zeilen werden umgebrochen.

n	[*Adresse1*][,*Adresse2*]n
	
	Einlesen der nächsten Zeile in den Arbeitsbereich. Die aktuelle Zeile wird auf die Standardausgabe geschrieben und die nächste Zeile zur aktuellen gemacht. Es wird danach der nächste auf **n** folgende Befehl ausgeführt, anstatt am Anfang des Skripts fortzufahren.
	
	Beispiel
	In den **ms**-Makros steht eine Abschnittsüberschrift auf einer Zeile unter einem .NH-Makro. Um alle Überschriften anzuzeigen, können Sie das folgende Skript mit **sed -n** aufrufen:
	
	`/^\.NH/{`
	`n`
	`p`
	`}`

N	[*Adresse1*][,*Adresse2*]N
	
	Anhängen der nächsten Eingabezeile an den Inhalt des Arbeitsbereiches. Die beiden Zeilen werden durch einen eingebetteten Zeilenwechsel getrennt. (Dieser Befehl ist gedacht für Mustervergleiche, die sich über mehrere Zeilen erstrecken.) Mit **\n** können Zeilenwechsel in Suchmustern angegeben werden, so daß es möglich ist, Mustervergleiche anzustellen, die sich über mehrere Zeilen erstrecken. Siehe dazu das Beispiel bei **D**.
	
	Beispiele
	Wie im vorigen Beispiel, aber die Zeile mit .NH wird zusätzlich zur Abschnittsüberschrift angegeben.
	
	`/^\.NH/{`
	`N`
	`p`
	`}`
	
	Zusammenfassen zweier Zeilen (Zeilenwechsel wird durch Leerzeichen ersetzt).
	
	`/^\.NH/{`
	`N`
	`s/\n/ /`
	`p`
	`}`

p	[*Adresse1*][,*Adresse2*]p Ausgeben der durch *Adresse* bestimmten Zeilen. Wenn nicht die Option **-n** auf der Kommandozeile angegeben wurde, führt dieser Befehl zu doppelten Zeilen in der Ausgabe. Typischerweise wird dieser Befehl vor Befehlen verwendet, welche die Abarbeitungsreihenfolge des Skripts beeinflussen (**d**, **N**, **b**) und so dazu führen

können, daß die aktuelle Zeile nicht ausgegeben wird. Siehe dazu die Beispiele bei **h**, **n** und **N**. | p

[*Adresse1*][,*Adresse2*]P | P

Ausgeben des ersten Teils bis zum ersten Zeilenwechsel eines mehrzeiligen Textes, der durch **N** angelegt wurde. Wenn auf die gleiche Zeile nicht vorher **N** angewendet wurde, ist **p** identisch mit **P**.

[*Adresse*]q | q

Abbrechen der Skriptabarbeitung, wenn die Adresse *Adresse* erreicht wird. Die angegebene Zeile wird zunächst ausgegeben (es sei denn, die Default-Ausgabe wurde unterdrückt), anschließend folgt eventuell mit **a** oder **r** angehängter Text.

Beispiele

Alles nach der bezeichneten Zeile löschen:

```
/Es folgt der kaputte Text:/q
```

Nur die ersten fünfzig Zeilen einer Datei ausgeben:

```
50q
```

[*Adresse*]r *Datei* | r

Einlesen des Inhalts von *Datei*. Der Dateiinhalt wird an den Inhalt des Arbeitsbereiches angehängt. Zwischen dem **r** und dem Dateinamen muß genau ein Leerzeichen stehen.

Beispiel

```
/Es folgt die Liste der Positionen:/r pos_datei
```

[*Adresse1*][,*Adresse2*]s/*Suchmuster*/*Ersetzungsmuster*/[*flags*] | s

Ersetzen von *Suchmuster* durch *Ersetzungsmuster* auf jeder durch *Adresse* bezeichneten Zeile. Wenn durch Muster spezifizierte Adressen verwendet wurden, bezeichnet das Muster **//** die letzte bezeichnete Musteradresse. Zusätzlich können die folgenden Optionen verwendet werden:

- *n* Ersetzen des *n*-ten Vorkommens von */Suchmuster/* auf jeder adressierten Zeile. *n* kann jede beliebige ganze Zahl zwischen 1 und 512 sein; der Default ist 1.
- g Ersetzen aller Vorkommen von */Suchmuster/* auf einer Zeile, nicht nur des ersten.
- p Ausgeben jeder Zeile, in der eine Ersetzung vorgenommen wurde. Wenn mehrere Ersetzungen in einer Zeile vorgenommen wurden, wird die Zeile auch mehrfach ausgegeben.
- w *Datei*
  Ausgeben jeder Zeile, in der eine Ersetzung vorgenommen wurde, in die Datei *Datei*.

→

s
←

Beispiele

Einige kurze, kommentierte Skripten:

```
# Ersetzen des 3. und 4. doppelten Anführungszeichens durch ( und ):
/function/{
s/"/(/3
s/"/)/4
}

# Entfernen aller doppelten Anfuehrungen auf einer bestimmten Zeile:
/Title/s/"//g

# Entfernen des ersten Doppelpunktes oder aller doppelten
#Anfuehrungszeichen; Ausgabe der resultierenden Zeilen:
s/://p
s/"//gp

# Ersetzen des ersten "if", das kein "ifdef" ist:
/ifdef/!s/if/   if/
```

t

[*Adresse1*][,*Adresse2*]t [*Sprungziel*]

Überprüfen, ob auf den durch *Adresse* bezeichneten Zeilen Ersetzungen vorgenommen worden sind. Wenn dies der Fall ist, Verzweigen auf die durch *:Sprungziel* markierte Zeile (siehe auch **b** und :). Wenn *Sprungziel* weggelassen wird, werden alle folgenden Befehle übersprungen. Der **t**-Befehl entspricht der **case**-Anweisung in der Programmiersprache C oder den Shell-Programmiersprachen. Jeder Fall wird überprüft; wenn eine Überprüfung wahr ist, wird das Konstrukt beendet.

Beispiel

Angenommen, Sie wollen leere Felder einer Datenbank auffüllen und haben die folgenden Daten:

```
ID: 1   Name: kalle   Gebiet: 45
ID: 2   Name: tanja
ID: 3
```

Sie wollen das folgende Format erreichen:

```
ID: 1   Name: kalle   Gebiet: 45   Telefon: ??
ID: 2   Name: tanja   Gebiet: ??   Telefon: ??
ID: 3   Name: ??      Gebiet: ??   Telefon: ??
```

Sie müssen also überprüfen, wie viele Felder bereits vorhanden sind. Dies kann mit dem folgenden Skript geschehen (es wird angenommen, daß die einzelnen Felder durch Tabulatoren getrennt sind):

```
/ID/{
s/ID: .* Name: .* Gebiet: .*/&   Telefon: ??/p
t
s/ID: .* Name: .*/&   Gebiet: ??   Telefon: ??/p
t
s/ID: .*/&   Name: ??     Gebiet: ??   Telefon: ??/p
}
```

[*Adresse1*][,*Adresse2*]w *Datei*

w

Anhängen des Inhalts des Arbeitsbereichs an die *Datei*. Das Speichern erfolgt in dem Moment, in dem der Befehl erreicht wird, nicht etwa erst, wenn der Arbeitsbereich wieder ausgegeben wird. Zwischen dem **w** und dem Dateinamen muß genau ein Leerzeichen stehen. Wenn die Datei noch nicht existiert, wird sie angelegt. Wenn sie bereits existiert, wird sie bei jeder Ausführung des Skripts überschrieben. Mehrfache **w**-Befehle, die in dieselbe Datei speichern, hängen ihre Daten jeweils hinten an die Datei an.

Beispiel

```
# Abspeichern von tbl- und eqn-Bloecken in eine Datei:
/^\.TS/,/^\.TE/w troff_kram
/^\.EQ/,/^\.EN/w troff_kram
```

[*Adresse1*][,*Adresse2*]x

x

Vertauschen des Arbeitsbereichs mit der Zwischenablage. Siehe das Beispiel bei der Beschreibung des **h**-Befehls.

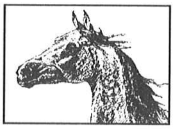
[*Adresse1*][,*Adresse2*]y/*abc*/*xyz*/

y

Konvertieren von Zeichen. Jedes Vorkommen von *a* wird in *x*, jedes *b* in *y*, jedes *c* in *z* konvertiert etc.

Beispiel

```
# 1, 2, 3, ... in A, B, C, ... umwandeln
/^item [1-9]/y/123456789/ABCDEFGHI/
```



KAPITEL 10

Die Skriptsprache gawk

In diesem Kapitel werden die folgenden Themen besprochen:

- Überblick über gawk
- Kommandozeilensyntax
- Muster und Prozeduren
- Systemvariablen
- Operatoren
- Variablen- und Feldzuweisungen
- Nach Gruppen sortierte Aufstellung der Befehle
- Alphabetisch sortierte Aufstellung der Befehle

Weitere Informationen finden Sie in *sed & awk*, zweite Auflage, von Dale Dougherty und Arnold Robbins.

Überblick über gawk

gawk ist die GNU-Version von **awk**, einem mächtigen Programm für Mustervergleiche bei der Verarbeitung von Textdateien, die aus Datensätzen fester oder variabler Länge bestehen können, die durch ein Trennzeichen (defaultmäßig der Zeilenwechsel) voneinander getrennt sind. **gawk** kann von der Kommandozeile aus oder in **gawk**-Skripten benutzt werden.

Mit **gawk** können Sie:

- Bequem eine Textdatei so verarbeiten, als bestünde sie aus Datensätzen und Feldern in einer textuellen Datenbank.
- Variablen benutzen, um die Datenbank zu verändern.
- Shell-Befehle aus Skripten heraus aufrufen.

- Arithmetische und Zeichenketten-Operationen ausführen.
- Programmiersprachenkonstrukte wie Schleifen und bedingte Ausdrücke verwenden.
- Eigene Funktionen definieren.
- Die Ergebnisse von Shell-Befehlen weiterverarbeiten.
- Kommandozeilenargumente benutzerfreundlicher verarbeiten.
- Formatierte Berichte erstellen.

Kommandozeilensyntax

Mit dem Aufruf von **awk** wird auf vielen Systemen, auf denen **gawk** vorhanden ist, letzteres aufgerufen. Es gibt zwei Aufrufformen für **gawk**:

> **gawk** [*Optionen*] '*Skript*' *Variable*=*Wert* *Datei(en)*
> **gawk** [*Optionen*] **-f** *Skriptdatei* *Variable*=*Wert* *Datei(en)*

Sie können entweder ein *Skript* direkt auf der Kommandozeile angeben oder ein Skript in einer Skriptdatei *Skriptdatei* ablegen und den Namen der Datei mit **-f** angeben. Wenn Sie sehr lange **gawk**-Programme verwenden, ist die Verwendung einer Skriptdatei vorzuziehen, weil Sie so die Fehlermeldung `arg list too long` der Shell vermeiden können. Variablen kann auf der Kommandozeile ein Wert zugewiesen werden. Der Wert kann ein Literal, eine Shellvariable (**$***name*) oder eine Befehls-Substitution sein, der Wert steht Ihrem **gawk**-Programm aber erst nach dem Lesen der ersten Eingabezeile (d.h. nach der **BEGIN**-Anweisung) zur Verfügung. **gawk** arbeitet auf einer oder mehreren Dateien. Wenn keine Dateien oder der Dateiname **-** angegeben werden, liest **gawk** von der Standardeingabe.

Um die ersten drei durch Doppelpunkte getrennten Felder jeweils auf einer eigenen Zeile auszugeben, können Sie folgenden Befehl verwenden:

```
gawk -F: '{print $1; print $2; print $3}' /etc/passwd
```

Sie finden viele verschiedene Beispiele für **gawk**-Skripten im Abschnitt „Muster und Prozeduren".

Optionen

Alle Optionen gibt es sowohl im traditionellen einbuchstabigen POSIX-Format als auch im langen GNU-Format. Hier einige der Optionen:

--
: Alle folgenden Argumente sind Befehle oder Dateinamen, aber keine Optionen.

-f *Datei*, **--file-program=***Datei*
: **gawk**-Befehle aus der *Datei* anstatt von der Kommandozeile lesen.

-v *Variable*=*Wert*, **--assign=***Variable*=*Wert*
: Zuweisung von *Wert* an *Variable*. Auf diese Art und Weise können Zuweisungen vorgenommen werden, die bereits vor Ausführung des Skripts gültig sind.

-F*c*, **--field-separator=***c*
: Einstellen des Feldseparators auf das Zeichen *c*. Der gleiche Effekt kann durch Setzen der Variablen **FS** erzielt werden. *c* darf ein regulärer Ausdruck sein. Jede Eingabezeile (jeder Datensatz) wird durch Leer- oder

Tabulatorzeichen oder einen benutzerdefinierten Feldseparator in Felder aufgeteilt. Auf die einzelnen Felder kann mit den Variablen **$1**, **$2** ... **$n**. zugegriffen werden. **$0** steht für die gesamte Eingabezeile.

-W *Option*

Alle -**W**-Optionen sind **gawk**-spezifisch, können also mit **awk** nicht verwendet werden. Alternativ zu dieser Schreibweise kann auch - -*Option* (z.B. - -**compat**) geschrieben werden. *Option* kann folgende Werte annehmen:

compat Genauso verhalten wie **awk**.

lint Warnungen bei Befehlen ausgeben, die sich nicht auf andere Versionen von **awk** übertragen lassen oder die **gawk** für problematisch hält.

posix Genaue Einhaltung der POSIX-Standards erwarten. Außerdem werden *x*-Escapesequenzen, die Synonymfunktion, ** und **= ignoriert.

source=*Programm*

Betrachte den Rest der Kommandozeile als **gawk**-Befehle, als ob sie aus einer Datei eingelesen würden.

gawk

Muster und Prozeduren

gawk-Skripten bestehen aus Mustern und Prozeduren:

Muster {*Prozedur*}

Beide können weggelassen werden. Wenn *Muster* fehlt, wird {*Prozedur*} auf alle Zeilen angewendet. Wenn {*Prozedur*} nicht angegeben wird, werden die passenden Zeilen ausgegeben.

Muster

Ein Muster kann eines der folgenden Elemente sein:

/regulärer Ausdruck/
relationaler Ausdruck
Mustervergleichsausdruck
Muster,Muster
BEGIN
END

- Ausdrücke können aus gequoteten Zeichenketten, Zahlen, Operatoren, Funktionen, definierten Variablen oder allen in *„gawk-Systemvariablen“* beschriebenen Variablen zusammengesetzt sein.
- Reguläre Ausdrücke benutzen eine erweiterte Menge von Metazeichen und werden in Kapitel 5 beschrieben.
- Außerdem können ^ und $ verwendet werden, um den Anfang bzw. das Ende eines Feldes zu bezeichnen (anstelle des Zeilenanfangs bzw. Zeilenendes wie normalerweise).
- Relationale Ausdrücke verwenden die relationalen Operatoren, die unter *„Operatoren“* näher beschrieben werden. Vergleiche können zeichenketten- oder zahlenbasiert sein. Beispielsweise werden mit **$2 > $1** alle Zeilen ausgewählt, in denen das zweite Feld größer als das erste ist.

- Mustervergleichsausdrücke verwenden die Operatoren ~ (Vergleich paßt) und !~ (Vergleich paßt nicht). Siehe dazu auch den Abschnitt *„Operatoren“* weiter hinten.
- Mit **BEGIN** können Prozeduren angegeben werden, die ausgeführt werden, *bevor* die erste Eingabezeile ausgeführt wird. In diesen Prozeduren werden typischerweise globale Variablen gesetzt.
- Mit **END** können Prozeduren angegeben werden, die ausgeführt werden, *nachdem* die letzte Eingabezeile gelesen wurde.
- Wenn es mehrere **BEGIN**- oder **END**-Muster gibt, werden die zugehörigen Aktionen in der Reihenfolge ausgeführt, in der sie im Skript auftauchen.
- *Muster*, *Muster* gibt einen Bereich von Zeilen an.

Mit Ausnahme von **BEGIN** und **END** können Muster mit den Booleschen Operatoren **||** (oder), **&&** (und) und **!** (nicht) verknüpft werden.

Prozeduren

Prozeduren bestehen aus einem oder mehren Befehlen, Funktionen oder Variablenzuweisungen, die durch Zeilenwechsel oder Semikola getrennt werden und von geschweiften Klammern umschlossen werden. Die Befehle können in vier Gruppen aufgeteilt werden:

- Variablen- oder Feldzuweisungen
- Ausgabebefehle
- Eingebaute Funktionen
- Kontrollflußbefehle

Einfache Beispiele mit Mustern und Prozeduren

1. Ausgeben des ersten Feldes jeder Zeile (kein Muster angegeben):

   ```
   { print $1 }
   ```

2. Alle Zeilen ausgeben, die *Muster* enthalten:

   ```
   /Muster/
   ```

3. Ausgeben des ersten Feldes von Zeilen, die *Muster* enthalten:

   ```
   /Muster/{ print $1 }
   ```

4. Ausgeben von Datensätzen mit mehr als zwei Feldern:

   ```
   NF > 2
   ```

5. Interpretieren von Eingabedatensätzen als Gruppe von Zeilen bis zur nächsten Leerzeile:

   ```
   BEGIN { FS = "\n"; RS = "" }
   ```

6. Ausgeben von Feld 2 und 3 in umgekehrter Reihenfolge, aber nur von den Zeilen, deren erstes Feld auf die Zeichenkette „DRINGEND“ paßt:

   ```
   $1 ~ /DRINGEND/ { print $3, $2 }
   ```

7. Zählen und ausgeben, wie oft *Muster* auftritt:

```
/Muster/ { ++x } END { print x }
```

8. Die Zahlen der zweiten Spalte aufaddieren und die Gesamtsumme ausgeben:

```
{total += $2 }; END { print "Die Spaltensumme ist", total}
```

9. Ausgeben von Zeilen, die weniger als 20 Zeichen enthalten:

```
length($0) < 20
```

10. Ausgeben jeder Zeile, die *Name:* enthält und genau sieben Felder hat:

```
NF == 7 && /^Name:/
```

11. Invertieren der Feldreihenfolge:

```
{ for (i = NF; i >= 1; i--) print $i }
```

gawk

gawk-Systemvariablen

Variable	Beschreibung
$*n*	*n*-tes Feld des aktuellen Datensatzes; die Felder sind durch **FS** getrennt
$0	Gesamter Datensatz
ARGC	Anzahl der Argumente der Kommandozeile
ARGIND	Stelle der aktuellen Datei auf der Kommandozeile (beginnend mit 0)
ARGV	Ein Feld mit den Kommandozeilenargumenten
CONVFMT	Konvertierungsformat für Zahlen (der Default ist %.6g)
ENVIRON	Ein assoziatives Feld, das die Umgebungsvariablen enthält
ERRNO	Beschreibung des letzten Systemfehlers
FIELDWIDTHS	Durch Leerzeichen getrennte Liste der Feldbreiten
FILENAME	Aktueller Dateiname
FNR	Wie **NR**, aber abhängig von der aktuellen Datei
FS	Feldseparator (der Default sind Leerzeichen, Tabulatoren und Zeilenwechsel)
IGNORECASE	Wenn diese Variable auf wahr gesetzt ist, wird bei Vergleichen keine Groß-/Kleinschreibung beachtet.
NF	Anzahl der Felder im aktuellen Datensatz
NR	Nummer des aktuellen Datensatzes
OFMT	Ausgabeformat für Zahlen (der Default ist %.6g)
OFS	Separator für Ausgabefelder (Default: ein Leerzeichen)
ORS	Separator für Ausgabedatensätze (Default: ein Zeilenwechsel)
RLENGTH	Länge des Strings, der auf die **match**-Funktion paßte
RS	Separator für Datensätze (Default: ein Zeilenwechsel)
RSTART	Erste Position in der Zeichenkette, die auf die **match**-Funktion paßte
SUBSEP	Trennzeichen für Feldindizes (der Default ist \034)

Operatoren

Die folgende Tabelle gibt alle in **gawk** verfügbaren Operatoren in der Reihenfolge aufsteigender Präzedenz an:

Symbol	Bedeutung
`= =+ -= *= /= %= ^=`	Zuweisung
`?:`	Bedingter Ausdruck wie in C
`||`	Logisches Oder
`&&`	Logisches Und
`~ !~`	Mustervergleich und negativer Mustervergleich
`< <= > >= != ==`	Relationale Operatoren
(leer)	Konkatenation
`+ -`	Addition und Subtraktion
`* / %`	Multiplikation, Division und Restbildung
`+ - !`	Unäres Plus und Minus und logische Negation
`^`	Potenzierung
`++ --`	Inkrementierung und Dekrementierung, entweder voran- oder nachgestellt
`$`	Feldreferenz

Variablen- und Feldzuweisungen

Variablen kann mit dem =-Zeichen ein Wert zugewiesen werden. Beispiel:

```
FS = ","
```

Außerdem können Ausdrücke mit den Operatoren +, -, *, / und % (Restbildung) an Variablen zugewiesen werden.

Felder werden mit der **split**-Funktion (siehe den Eintrag in „*Alphabetische Zusammenfassung der Befehle*") angelegt oder einfach in einer Zuweisungsanweisung angesprochen. Feldelemente werden mit Zahlen (*Feld*[1] ... *Feld*[*n*]) oder mit Namen indiziert. So kann beispielsweise die Anzahl der Vorkommen eines Musters mit folgendem Skript gezählt werden:

```
/Muster/ { Feld["/Muster/"]++ }
END { print Feld["/Muster/"] }
```

In **gawk** müssen Variablen vor ihrer Verwendung nicht deklariert und Felder nicht dimensioniert werden; sie werden jeweils beim ersten Zugriff angelegt. Alle Variablen werden als Zeichenketten abgelegt, können aber als Zeichenketten oder als Zahlen verwendet werden. **gawk** ermittelt aus dem Kontext, ob eine Variable als Zeichenkette oder als Zahl interpretiert werden muß, dies kann aber auch vom Benutzer festgelegt werden. Um zu erzwingen, daß eine Variable als Zeichenkette interpretiert wird, muß eine leere Zeichenkette angehängt werden:

```
Variable ""
```

Um zu erzwingen, daß eine Variable als Zahl interpretiert wird, muß Null hinzuaddiert werden:

```
Variable + 0
```

Nach Gruppen sortierte Aufstellung der gawk-Befehle

gawk-Befehle können folgendermaßen eingeteilt werden:

Arithmetische Funktionen	Zeichenketten-Funktionen	Kontrollfluß-Anweisungen	Ein-/Ausgabe-Verarbeitung	Zeit-Funktionen	Diverses
atan2	gsub	break	close	systime	delete
cos	index	continue	getline	strftime	function
exp	length	do/while	next		system
int	match	exit	print		
log	split	for	printf		
rand	sub	if	sprintf		
sin	substr	return			
sqrt	tolower	while			
srand	toupper				

gawk

Alphabetisch sortierte Zusammenfassung der Befehle

Die folgende Liste enthält alle in **gawk** unter Linux zur Verfügung stehenden Befehle und Funktionen:

atan2 (*y*,*x*)

atan2

Gibt den Arcus Tangens von *y*/*x* im Bogenmaß zurück.

break

break

Abbrechen einer **while**- oder **for**-Schleife.

close (*Dateiname=Ausdr*)
close (*Befehls-Ausdr*)

close

In den meisten **gawk**-Implementierungen können nur zehn Dateien und eine Pipe geöffnet sein. Daher stellt **gawk** die **close**-Anweisung zur Verfügung, mit der eine Datei oder eine Pipe wieder geschlossen werden kann. Das Argument ist das gleiche wie beim Öffnen der Datei oder der Pipe.

continue

continue

Unmittelbar den nächsten Schleifendurchlauf einer **while**- oder **for**-Schleife anfangen, ohne die restlichen Befehle bis zum Schleifenende abzuwarten.

cos (*x*)

cos

Gibt den Cosinus von *x* (ein Winkel im Bogenmaß) zurück.

delete (*Feld*[*Element*])

delete

Löscht ein *Element* in *Feld*.

do	do *Schleifenkörper* **while** (*Ausdr*) Schleifenkonstrukt: Die Anweisungen im *Schleifenkörper* werden ausgeführt. Danach wird *Ausdr* evaluiert. Ist das Ergebnis wahr, wird der *Schleifenkörper* erneut ausgeführt.
exit	exit Keine weiteren Anweisungen ausführen und keine weiteren Eingabezeilen lesen. END-Prozeduren werden aber noch ausgeführt.
exp	exp (*arg*) Gibt den Wert e^*arg* zurück. (Diese Funktion ist die Inversion von **log**.)
for	for (*i=von* ; *i<= bis* ; *i++*) *Befehl* Solange die Variable *i* zwischen *von* und *bis* liegt, wird *Befehl* ausgeführt. Wenn es sich bei *Befehl* um mehrere Befehle handelt, müssen diese in geschweiften Klammern zusammengefaßt sein. Anstelle von <= kann auch jeder andere Vergleichsoperator verwendet werden; außerdem ++ zum Inkrementieren und -- zum Dekrementieren der Variablen.
for	for (*Element* in *Feld*) *Befehl* Für jedes *Element* in dem assoziativen Feld *Feld* wird einmal *Befehl* ausgeführt. Wenn es sich bei *Befehl* um mehrere Befehle handelt, müssen diese in geschweiften Klammern zusammengefaßt sein. Auf jedes Element des Feldes kann mit *Feld*[*Element*] zugegriffen werden. Die Elemente von **gawk**-Feldern werden so abgelegt, daß auf jedes Element mit etwa der gleichen Geschwindigkeit zugegriffen werden kann. Die Reihenfolge erscheint dem Benutzer daher zufällig. Wenn die Ausgabe in einer sortierten Reihenfolge geschehen soll, muß sie noch den **sort**-Befehl durchlaufen.

function	function *Name*(*Parameter-Liste*) {
	Anweisungen
	}
	
	Anlegen der benutzerdefinierten Funktion *Name*, die aus den **gawk**-Befehlen *Anweisungen* besteht, denen die angegebenen Parameter zur Verfügung stehen.

getline	getline [*Variable*] [<*Datei*] *Befehl* | getline [*Variable*] Die erste Variante liest Eingabedaten aus *Datei* oder der nächsten Datei auf der Kommandozeile; die zweite liest die Ausgabe von *Befehl* als Eingabe. Beide

Varianten lesen eine Zeile zur gleichen Zeit und pro Aufruf. Die Eingabezeile wird der Variablen $0 zugewiesen und in Felder zerlegt, wobei die Variablen **NF**, **NR** und **FNR** entsprechend gesetzt werden. Wenn *Variable* angegeben wird, wird die Eingabezeile statt nach $0 nach *Variable* geschrieben; $0 bleibt in diesem Fall unverändert. In diesem Fall wird also auch die aktuelle Zeile nicht verändert. **getline** ist eine Funktion und gibt 1 zurück, wenn ein Datensatz erfolgreich gelesen werden konnte, 0, wenn die Eingabedatei zu Ende ist, und -1, wenn ein anderer Fehler aufgetreten ist. getline

gsub (*r,s,t*) gsub

Globales Ersetzen des regulären Ausdrucks *r* durch *s* in der Zeichenkette *t*. **gsub** gibt die Anzahl der durchgeführten Ersetzungen zurück. Wenn *t* nicht angegeben wird, wird $0 verwendet.

if (*Bedingung*) if
    Befehl1
[else]
    [*Befehl2*]

Wenn die *Bedingung* wahr ist, wird der Befehl *Bedingung1* ausgeführt, ansonsten *Bedingung2*. Die Bedingung kann mit Ausdrücken formuliert werden, die einen der Vergleichsoperatoren <, <=, ==, !=, >= oder > oder den Mustervergleichsoperator ~ enthalten. Wenn es sich bei *Bedingung1* oder *Bedingung2* um mehrere Befehle handelt, müssen diese in geschweiften Klammern stehen.

Beispiel

Mit der folgenden Programmzeile wird bestimmt, ob das erste Wort jeder Zeile mit einem kleinen oder großen A anfängt:

```
if ($1 ~ /[Aa]*/)
```
...fängt mit A oder a an

index (*Substr,Str*) index

Gibt die Position der Teilzeichenkette *Substr* in der Zeichenkette *Str* zurück.

int (*Arg*) int

Gibt den ganzzahligen Anteil von *Arg* zurück.

length (*Arg*) length

Gibt die Länge von *Arg* zurück. Wenn *Arg* nicht angegeben ist, wird $0 angenommen. **length** kann daher auch als vordefinierte Variable angesehen werden, welche die Länge des aktuellen Datensatzes enthält.

log (*Arg*) log

Gibt den Logarithmus naturalis von *Arg* zurück. Diese Funktion ist die Inversion von **exp**.

match	**match** (*s*,*r*) Gibt die erste Position in *s* zurück, auf die der reguläre Ausdruck *r* paßt. Wenn *r* nirgendwo paßt, wird 0 zurückgegeben. **match** setzt die Variablen **RSTART** und **RLENGTH**.
next	**next** Lesen der nächsten Eingabezeile und Anfangen eines neuen Durchlaufs durch die Muster-/Prozedur-Anweisungen.
print	**print** [*Args*] [*Ausgabeziel*] Gibt *Args* auf den Ausgabekanal aus. Wörtlich auszugebende Zeichenketten müssen gequotet werden. Datenfelder werden in der Reihenfolge, in der sie angegeben werden, ausgegeben. Wenn sie durch Kommata in der Argumentliste getrennt sind, werden sie bei der Ausgabe durch das mit **OFS** festgelegte Zeichen getrennt; wenn sie durch Leerzeichen getrennt sind, werden sie bei der Ausgabe aneinandergehängt. Bei *Ausgabeziel* handelt es sich um einen Shell-Ausdruck zur Ausgabeumleitung, der bestimmt, wohin die Daten geschrieben werden.
printf	**printf** [*Format* [, *Ausdrücke*]] Formatierte **print**-Ausgabe. Ausdrücke und Variablen können entsprechend den Anweisungen im *Format*-Argument formatiert werden. Die Anzahl der Ausdrücke muß mit der in der Formatanweisung angegebenen Zahl übereinstimmen. *Format* verwendet dieselben Konventionen wie die **printf**-Funktion der Programmiersprache C. Im folgenden die gebräuchlichsten Formate: %s — Eine Zeichenkette. %d — Eine ganze Dezimalzahl. %*n*.*m*f — Eine Fließkommazahl; *n* gibt die Gesamtanzahl auszugebender Stellen, *m* die Anzahl auszugebender Stellen nach dem Dezimaltrennzeichen an. %[-]*nc* — *n* gibt die minimale Feldlänge für eine Formatierung des Typs *c* an. Wenn – angegeben ist, wird der Wert bei der Ausgabe zentriert, ansonsten rechtsbündig ausgerichtet. Zusätzlich kann eine Feldbreite angegeben werden. *Format* darf auch eingebettete Escape-Sequenzen enthalten; die gebräuchlichsten sind \n (Zeilenwechsel) und \t (Tabulator). Leerzeichen und wörtlicher Text dürfen im Format-Argument stehen, wenn das gesamte Argument gequotet wird. Wenn mehrere Ausdrücke ausgegeben werden sollen, müssen auch mehrere Formatanweisungen vorhanden sein.

Beispiel

Mit dem Skript:

```
{printf ("Die Summe der Zeile %s ist %d.\n", NR, $1+$2)}
```

und der Eingabedatei:

```
5   5
```

wird der folgende Text, gefolgt von einem Zeilenwechsel, ausgegeben:

```
The sum on line 1 is 10.
```

printf

rand ()

rand

Erzeugen einer Zufallszahl zwischen 0 und 1. Diese Funktion gibt bei jedem Aufruf des Skripts die gleiche Folge von Zufallszahlen zurück, es sei denn, dem Zufallszahlengenerator wird mit der **srand()**-Funktion ein Startwert vorgegeben.

return [*Ausdr*]

return

Gibt in benutzerdefinierten Funktionen den Wert *Ausdr* zurück und beendet die Funktion.

sin (*x*)

sin

Gibt den Sinus von *x* (ein Winkel im Bogenmaß) zurück.

split (*String*,*Feld*[,*Sep*])

split

Aufbrechen der Zeichenkette *String* in die Feldelemente *Feld*[1] ... *Feld*[*n*]. Die Zeichenkette wird bei jedem Vorkommen des Separators *Sep* aufgebrochen. Wenn *Sep* nicht angegeben wird, wird der Wert von **FS** verwendet. **split** gibt die Anzahl der angelegten Feldelemente zurück.

sprintf [*Format* [, *Ausdrücke*]]

sprintf

Gibt den Wert eines oder mehrerer *Ausdrücke* nach den Anweisungen in *Format* formatiert zurück; die Daten werden lediglich formatiert, aber nicht ausgegeben.

sqrt (*Arg*)

sqrt

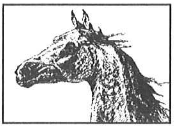
Gibt die Quadratwurzel von *Arg* zurück.

srand (*Ausdr*)

srand

Setzen eines neuen Startwertes für den Zufallszahlengenerator. Der Default für *Ausdr* ist die aktuelle Tageszeit.

strftime	**strftime** (*Format, Zeit*) Gibt die *Zeit* (die genauso formatiert sein sollte, wie sie von **systime()** zurückgegeben wird) im durch *Format* spezifizierten Format zurück. Der Aufbau von *Format* ist derselbe wie bei der C-Funktion **strftime** oder dem in Kapitel 1 besprochenen **date**-Befehl.
sub	**sub** (*r,s,t*) Ersetzen der ersten Stelle in *t*, auf die der reguläre Ausdruck *r* paßt, durch *s*. Wenn ersetzt werden konnte, wird 1 zurückgegeben, ansonsten 0. Der Default für *t* ist $0.
substr	**substr** (*String,m*[*,n*]) Ausgeben der Teilzeichenkette von *String*, die an Position *m* beginnt und aus den folgenden *n* Zeichen besteht. Wenn *n* nicht angegeben wird, werden alle folgenden Zeichen bis zum Ende der Zeichenkette zurückgegeben.
system	**system** (*Befehl*) Ausführen des angegebenen Shell-Befehls. Der Rückgabewert dieses Befehls ist auch der Rückgabewert von **system()**. Üblicherweise steht dabei 1 für eine erfolgreiche Ausführung, 0 für vollständige Ausführung (z.B. Dateiende) und –1 für einen Fehler. Die Ausgabe des Shell-Befehls steht dem **gawk**-Skript nicht zur Verfügung. Hierzu muß die Syntax „*Befehl* **|** **getline**" verwendet werden.
systime	**systime** () Gibt die Anzahl der Sekunden seit Mitternacht des 1. Januar 1970 zurück.
tolower	**tolower** (*String*) Ändert alle Großbuchstaben in *String* in Kleinbuchstaben und gibt die geänderte Zeichenkette zurück.
toupper	**toupper** (*String*) Ändert alle Kleinbuchstaben in *String* in Großbuchstaben und gibt die geänderte Zeichenkette zurück.
while	**while** (*Bedingung*) *Befehl* Führt *Befehl* aus, solange die *Bedingung* wahr ist. Eine Beschreibung der erlaubten Bedingungsausdrücke finden Sie in der Beschreibung der **if**-Anweisung. Wenn es sich bei *Befehl* um mehrere Befehle handelt, müssen diese von geschweiften Klammern umschlossen sein.



KAPITEL 11

Perl-Schnellübersicht

Dieses Kapitel enthält eine Schnellübersicht der Syntax sowie der Funktionen von Perl. Für einen gründlichen Einstieg in Perl empfehlen wir *Einführung in Perl* von Randal Schwartz. Eine ausführliche Perl-Referenz finden Sie in *Programmieren mit Perl* von Larry Wall, Tom Christiansen und Randal Schwartz.

Alle Elemente, denen ein Dolchzeichen (†) folgt, ergeben sich zu $_, wenn sie ausgelassen werden.

Optionen auf der Befehlszeile

-a Aktiviert zusammen mit -n oder -p den Autosplit-Modus. Es trennt dabei bei @F.

-c Überprüft nur die Syntax, führt das Programm aber nicht aus.

-d Führt das Skript unter dem Debugger aus. Verwenden Sie -de 0, um den Debugger ohne ein Skript zu starten.

-D *zahl*
Setzt Debug-Flags.

-e *befehlszeile*
Kann verwendet werden, um eine einzelne Zeile zu übergeben. Die Option -e kann auch mehrfach eingesetzt werden, um ein mehrzeiliges Skript aufzubauen.

-F *reg_ausdruck*
Spezifiziert einen regulären Ausdruck zum Splitten, wenn -a aktiv ist.

-i *erweiterung*
Editiert Dateien, die vom Konstrukt < > verarbeitet werden, an Ort und Stelle.

-I *verzeichnis*
Sagt dem durch -P aufgerufenen C-Präprozessor, wo Include-Dateien gesucht werden sollen. Dieses Verzeichnis wird @INC vorangestellt.

-l *oktalzahl*
Aktiviert die automatische Zeilenendeerkennung, beispielsweise -l013.

-n Umgibt Ihr Skript mit einer Eingabeschleife. Die Zeilen werden nicht ausgegeben.

-p Umgibt Ihr Skript mit einer Eingabeschleife. Die Zeilen werden ausgegeben.

-P Läßt das Skript durch den C-Präprozessor laufen, bevor es von Perl kompiliert wird.

-s Interpretiert -xxx auf der Befehlszeile als einen Schalter und setzt die Umgebungsvariable $xxx für das Skript.

-S Sucht das Skript im durch die Umgebungsvariable PATH definierten Pfad.

-T Erzwingt strenge Überprüfung.

-u Erzwingt einen Coredump, nachdem das Skript kompiliert wurde. Zur Verwendung mit *undump*(1) (sofern verfügbar).

-U Erlaubt Perl, unsichere Operationen durchzuführen.

-v Gibt Version und Patchlevel Ihres Perl-Executables aus.

-w Gibt Warnungen vor möglichen Schreibfehlern und anderen fehleranfälligen Konstrukten im Skript.

-x [*verzeichnis*]
Extrahiert das Perl-Programm aus der Standardeingabe. Wird ein *verzeichnis* angegeben, so wird vorher in dieses gewechselt.

-0 *wert*
Gibt den Anfangswert für den Datensatztrenner $/ vor. Siehe auch -l.

Literale

Numerisch
123 1_234 123.4 5E-10 0xff (hexadezimal) 0377 (oktal)

String
´abc´
Wörtlich zu übernehmender Text, keine Interpretation von Variablen oder Escapesequenzen außer \' und \\. Alternativ: q/abc/. Statt /.../ kann auch fast jedes andere Paar von Trennern verwendet werden.

"abc"
Variablen werden interpretiert und Escapesequenzen verarbeitet. Alternative: qq/abc/.

Escapesequenzen: \t (Tab), \n (Newline), \r (Return), \f (Formfeed), \a (Alarm), \b (Backspace), \e (Escape), \033 (Oktal), \x1b (hexadezimal), \c[(Control).

\l und \u wandeln das folgende Zeichen in Klein- bzw. Großschreibung um. \L und \U wandeln Zeichen in Klein- bzw. Großschreibung um, bis ein \E auftaucht. \Q quotet reguläre Ausdrücke, bis ein \E auftaucht.

\`*Kommando*\` wertet die Ausgabe von *Kommando* aus. Alternativ: qx/*Kommando*/.

Feld

(1, 2, 3)
() ist ein leeres Feld.
(1..4) ist äquivalent zu (1,2,3,4), auch ('a'..'z') ist möglich.
qw/foo bar .../ ist äquivalent zu ('foo','bar',...).

Feldreferenz

[1,2,3]

Hash (assoziatives Feld)

(*schlüssel1*, *wert1*, *schlüssel2*, *wert2*, ...)
Auch: (*schlüssel1* => *wert1*, *schlüssel2* => *wert2*, ...)

Hash-Referenz

{*schlüssel1*, *wert1*, *schlüssel2*, *wert2*, ...}

Codereferenz

sub { *Anweisungen* }

Dateideskriptoren

STDIN, STDOUT, STDERR, ARGV, DATA.
Benutzerdefiniert: *handle*, $*var*.

Globs <*Muster*>

Wird in alle Dateinamen gewandelt, auf die das Suchmuster paßt. Verwenden Sie <${*Var*}> oder `glob` $*Var*, um diesen Mechanismus mit einer Variable anzuwenden.

Direktreferenz

<<*Identifikator*
Direkt eingefügtes Dokument (wie bei der Shell).

Spezielle Token

_ _FILE_ _: Dateiname,
_ _LINE_ _: Zeilennummèr,
_ _END_ _: Ende des Programms, die restlichen Zeilen können unter dem Dateideskriptor

DATA eingelesen werden.

Variablen

$var

Eine einfache skalare Variable.

$var[28]

29. Element des Feldes @var.

$p = \@var

Deklariert $p als Referenz auf das Feld @var.

$$p[28]

29. Element des durch $p referenzierten Feldes. Alternativ: $p->[28].

$var[-1]
: Letztes Element des Feldes @var.

$var[$i][$j]
: $j-tes Element des $i-ten Elements des Feldes @var.

$var{'Feb'}
: Ein Wert des assoziativen (Hash-)Feldes %var.

$p=\%var
: Deklariert $p als Referenz auf das Hash-Feld %var.

$$p{'Feb'}
: Ein Element des durch $p referenzierten Hash-Feldes. Alternativ: $p->{'Feb'}.

$#Var
: Letzter Index des Feldes @Var.

@var
: Das ganze Feld, in skalarem Kontext die Anzahl der Elemente des Feldes.

@var[3,4,5]
: Ein Teil des Feldes @var.

@var['a','b']
: Ein Teil von %Var, gleichwertig zu ($var{'a'},$var{'b'}).

%var
: Ein komplettes Hash-Feld. Im skalaren Kontext ein TRUE, wenn Elemente existieren.

$var{'a',1,...}
: Emuliert ein mehrdimensionales Feld.

('a'...'z')[4,7,9]
: Ein Teil eines konstant definierten Feldes.

pkg::var
: Eine Variable einer Unterbibliothek, z.B. $pkg::var, @pkg::feld.

\ *objekt*
: Referenz auf ein Objekt, z.B. \$var, \%hash.

**name*
: Bezeichnet alle Objekte, die durch *name* dargestellt werden.
: *n1 = *n2 macht n1 zum Alias für n2.
: *n1 = $n2 macht $n1 zum Alias für $n2.

Sie können statt eines Variablenbezeichners grundsätzlich auch einen { *Block* } verwenden, der eine passende Referenz zurückliefert, z.B. ${...}, &{...}. $$p ist nur eine Abkürzung für ${$p}.

Operatoren

<table>
<tr><td><code>**</code></td><td>Potenzierung</td></tr>
<tr><td><code>+ - * /</code></td><td>Addition, Subtraktion, Multiplikation, Division</td></tr>
<tr><td><code>%</code></td><td>Modulodivision, ergibt Rest einer Ganzzahldivision</td></tr>
<tr><td><code>& | ^</code></td><td>Bitweises UND, bitweises ODER, bitweises Exklusiv-Oder.</td></tr>
<tr><td><code><< >></code></td><td>Bitweises Schieben nach links und nach rechts.</td></tr>
<tr><td><code>|| &&</code></td><td>Logisches ODER, logisches UND</td></tr>
<tr><td><code>.</code></td><td>Verknüpfung zweier Strings</td></tr>
<tr><td><code>x</code></td><td>Liefert einen String oder ein Feld, der oder das aus dem linken Operanden (String oder Feld) besteht, der oder das so oft wiederholt wird, wie es vom rechten Operanden angegeben wird.</td></tr>
</table>

Alle obigen Operatoren haben eine Variante mit einem Zuweisungsoperator, z.B. `.=`

`->`	Dereferenzierungsoperator
`\`	Unäre Referenz
`! ~`	Unäre Negation, unäres Bitkomplement
`++ --`	Auto-Inkrement, Auto-Dekrement
`== !=`	Numerische Gleichheit und Ungleichheit
`eq ne`	Stringgleichheit und -ungleichheit
`< >`	Numerisches Kleiner-als und Größer-als
`lt gt`	Kleiner-als und Größer-als für Strings
`<= >=`	Numerisches Kleiner-gleich bzw. Größer-gleich
`le ge`	Kleiner-gleich bzw. Größer-gleich für Strings
`<=> cmp`	Numerischer (String-) Vergleich. Gibt -1, 0 oder 1 aus.
`=~ !~`	Suchmuster, Mustersetzung oder zeichenweise Ersetzung
`..`	Bereich (skalarer Kontext) oder Aufzählung (Feldkontext)
`?:`	Alternativoperator (if-then-else)
`,`	Kommaoperator, auch Trenner für Listenelemente (auch `=>`)
`not`	Negation niedrigen Vorrangs
`and`	UND niedrigen Vorrangs
`or xor`	ODER/Exklusiv-ODER niedrigen Vorrangs

Alle Perl-Funktionen können auch als Listenoperatoren verwendet werden. In diesem Fall haben sie entweder sehr hohen oder sehr niedrigen Vorrang, je nachdem, ob Sie die linke oder die rechte Seite des Operators betrachten. Nur die Operatoren `not`, `and`, `or` und `xor` haben noch niedrigeren Vorrang.

Eine „Liste" ist eine Liste von Ausdrücken, Variablen oder Listen. Statt einer Liste kann immer auch eine Feldvariable oder ein Teil eines Feldes verwendet werden. Die Parameterlisten können dann in Klammern eingeschlossen werden, um Vorrangsprobleme zu vermeiden.

Anweisungen

Jede Anweisung ist ein Ausdruck, dem optional ein Modifikator folgt und der von einem Semikolon beendet wird. Das Semikolon kann entfallen, wenn es sich um die letzte Anweisung in einem Block handelt.

Die Ausführung von Ausdrücken kann von anderen Ausdrücken abhängen, wenn einer der Modifikatoren `if`, `unless`, `while` oder `until` verwendet wird, zum Beispiel:

```
Ausdruck1 if Ausdruck2;
Ausdruck1 until Ausdruck2;
```

Die logischen Operatoren `||`, `&&` oder `?:` erlauben auch die bedingte Ausführung:

```
Ausdruck1 || Ausdruck2;
Ausdruck1 ? Ausdruck2 : Ausdruck3;
```

Anweisungen können zu einem *Block* kombiniert werden, indem sie in geschweifte Klammern { } eingebettet werden. Blöcke können dann zur Ablaufsteuerung verwendet werden:

```
if (Ausdruck) Block [[ elsif (Ausdruck) Block ... ] else Block ]
unless (Ausdruck) Block [ else Block ]
[ label: ] while (Ausdruck) Block [ continue Block ]
[ label: ] until (Ausdruck) Block [ continue Block ]
[ label: ] for ([Ausdruck]; [ Ausdruck]; [ Ausdruck]) Block
[ label: ] foreach Var† (Liste) Block
[ label: ] Block [ continue Block ]
```

Der Programmablauf kann durch folgende Anweisungen gesteuert werden:

`goto` *label*
: Setzt die Ausführung am angegebenen Label fort.

`last` [*label*]
: Beendet die entsprechende Schleife unmittelbar. Überspringt den continue-Block.

`next` [*label*]
: Startet den nächsten Schleifendurchlauf.

`redo` [*label*]
: Startet den Schleifenblock neu, ohne die Schleifenbedingung neu auszuwerten.

Spezielle Formen sind:

```
do Block while Ausdruck;
do Block until Ausdruck;
```

bei denen *Block* garantiert ausgeführt wird, bevor *Ausdruck* überprüft wird, und

```
do Block
```

wodurch *Block* effektiv in einen Ausdruck überführt wird.

Unterroutinen, Pakete und Module

&subroutine Liste
: Führt eine durch eine `sub`-Deklaration definierte Unterroutine aus und gibt den Wert des letzten evaluierten Ausdrucks in der Unterroutine zurück. *Subroutine* kann auch ein Ausdruck sein, der eine Referenz auf ein Codeobjekt liefert. & kann entfallen, wenn die Deklaration der Unterroutine vor ihrer Verwendung erfolgte.

`bless` *Ref* [, *Paket*]
: Wandelt das referenzierte Objekt *Ref* in ein Objekt in *Paket* um und gibt die Referenz darauf zurück.

`caller` [*Ausdruck*]
: Gibt ein Feld ($package, $file, $line, ...) für einen bestimmten Unterroutinenaufruf zurück. `caller` liefert diese Information für die aktuelle Unterroutine, `caller(1)` für die Routine, in der diese aufgerufen wurde, und so fort. Gibt `false` zurück, wenn kein Aufrufer existiert.

`do` *Liste von Unterroutinen*
: Unübliche Form von *&subroutine.*

`goto` *&subroutine*
: Ersetzt die aktuelle Unterroutine durch einen Aufruf von *subroutine.*

`import` *Modul* [[*Version*] *Liste*]
: Importiert die angegebenen Unterroutinen aus Modul.

`no` *Modul Liste*
: Widerruft importierte Semantik. Siehe auch `use`.

`package` *Name*
: Deklariert den Rest des aktuellen Blocks als ein Paket.

`require` *Ausdr*†
: Wenn *Ausdr* numerisch ist, muß Perl mindestens die angegebene Version haben. Ansonsten muß *Ausdr* der Name einer Datei sein, die aus der Perl-Bibliothek eingefügt wird. Sie kann allerdings nicht mehrfach eingebunden werden, und es führt zu einer Fehlermeldung, wenn die Datei nicht zu `true` evaluiert. Wenn *Ausdr* ein Wort ohne Suffix ist, wird .pm als Extension für die Datei angenommen.

`return` *Ausdr*
: Kehrt mit dem angegebenen Wert aus einer Subroutine zurück.

sub *Name* { *Ausdr*; ... }
: Definiert *Name* als Unterroutine. Parameter werden durch Referenz übergeben als Feld @_. Gibt den Wert des letzten evaluierten Ausdrucks zurück.

[`sub`] BEGIN { *Ausdr*; ... }
: Definiert einen Initialisierungsblock, der vor der Ausführung aufgerufen wird.

[`sub`] END { *Ausdr*; ... }
: Definiert einen Beendigungsblock, der vor dem Beenden aufgerufen wird.

`tie` *Var, Paket, [Liste]*
: Bindet eine Variable an ein Paket, das für sie zuständig ist. Kann verwendet werden, um eine *dbm-* oder *ndbm*-Datei an ein Hash zu binden.

`untie` *Var*
: Hebt die Bindung zwischen einem Paket und einer Variable auf.

`use` *Modul* [[*Version*] *Liste*]
: Importiert Semantik vom angegebenen Modul in das aktuelle Paket.

Objektorientierte Programmierung

Perl hat einige Regeln für die objektorientierte Programmierung:

- Ein Objekt ist einfach nur eine Referenz, die zufällig weiß, zu welcher Klasse sie gehört. Objekte sind „erleuchtet", Referenzen nicht.
- Eine Klasse ist ein Paket, das Methoden zur Verfügung stellt, um Objektreferenzen zu behandeln. Wenn ein Paket keine Methode zur Verfügung stellt, werden die in @ISA aufgeführten Basisklassen durchsucht.
- Eine Methode ist eine Unterroutine, die als erstes Argument eine Objektreferenz erwartet (bei statischen Methoden einen Paketnamen).

Methoden können wie folgt angewendet werden:

Methode Objektref Parameter oder
Objektref->Methode Parameter

Arithmetische Funktionen

`abs` *Ausdr*†
: Gibt den absoluten Wert des Operanden zurück.

`atan2` *y, x*
: Gibt den Arkustangens von *y/x* im Bereich von -π bis π zurück.

`cos` *Ausdr*†
: Gibt den Kosinus von *Ausdr* zurück (im Bogenmaß).

`exp` *Ausdr*†
: Gibt e in der *Ausdr*-ten Potenz zurück.

`int` *Ausdr*†
: Gibt den Ganzzahlanteil von *Ausdr* zurück.

`log` *Ausdr*†
: Gibt den natürlichen Logarithmus (zur Basis e) von *Ausdr* zurück.

`rand` [*Ausdr*]
: Gibt eine zufällige rationale Zahl zwischen 0 und dem Wert von *Ausdr* zurück. Wenn *Ausdr* fehlt, wird ein Wert zwischen 0 und 1 zurückgegeben.

`sin` *Ausdr*†
: Gibt den Sinus von *Ausdr* zurück (im Bogenmaß).

`sqrt` *Ausdr*†
: Gibt die Quadratwurzel von *Ausdr* zurück.

`srand` [*Ausdr*]
: Setzt den Ausgangswert des Zufallszahlengenerators.

`time`
: Gibt die Anzahl von Sekunden seit dem 1. Januar 1970 zurück. Zur Übergabe an `gmtime` und `localtime` geeignet.

Umwandlungsfunktionen

`chr` *Ausdr*†
: Gibt das Zeichen zurück, das durch den Dezimalcode *Ausdr* repräsentiert wird.

`gmtime` *Ausdr*†
: Wandelt eine Zeit im von `time` übergebenen Format in ein Feld mit 9 Elementen um (0:$sec, 1:$min, 2:$hour, 3:$mday, 4:$mon, 5:$year, 6:$wday, 7:$yday, 8:$isdst), wobei GMT gilt. $mon liegt im Bereich von 0..11 und $wday im Bereich von 0..6.

`hex` *Ausdr*†
: Gibt den Dezimalwert des Hexstrings *Ausdr* zurück.

`localtime` *Ausdr*
: Wandelt die von `time` zurückgegebene Zeit in einen *ctime*(3)-String um. Im Arraykontext liefert `localtime` ein Array mit 9 Elementen, wobei die lokale Zeitzone gilt.

`oct` *Ausdr*†
: Gibt den Dezimalwert des Oktalstrings *Ausdr* zurück. Beginnt *Ausdr* mit 0x, wird der String statt dessen als Hexzahl interpretiert.

`ord` *Ausdr*†
: Gibt den ASCII-Wert des ersten Zeichens von *Ausdr* zurück.

`vec` *Ausdr, Offset, Bits*
: Behandelt den String *Ausdr* als Vektor aus vorzeichenlosen Ganzzahlen und gibt den Teil an der Stelle *Offset* zurück. *Bits* kann dabei Zweierpotenzen von 1..32 annehmen. Kann auch einen Wert zugewiesen bekommen.

Strukturumwandlungen

`pack` *template, Liste*
: Packt die übergebenen Werte in eine binäre Struktur, wobei die Schablone *template* verwendet wird.

`unpack` *template, Ausdr*
: Packt die Struktur *Ausdr* in einem Feld aus, wobei die Schablone *template* verwendet wird.

template ist in beiden Fällen eine Zeichenfolge aus den folgenden Zeichen:

a / A		ASCII-String, mit Null-/Leerzeichen aufgefüllt
b / B		Bitfolge in steigender/fallender Reihenfolge
c / C		Nativer/vorzeichenloser Zeichenwert
f / d		Fließkommazahl einfacher/doppelter Genauigkeit
h / H		Hexstring, beginnend mit niederwertigem/höherwertigem Nibble
i / I		Vorzeichenbehaftete/vorzeichenfreie Ganzzahl
l / L		Vorzeichenbehaftetes/vorzeichenfreies Langwort
n / N		Wort/Langwort in Big-Endian-Darstellung
s / S		Vorzeichenbehaftetes/vorzeichenfreies Wort
u / p		Mit UUencode kodierter String/Pointer auf einen String
v / V		Wort/Langwort in Little-Endian-Darstellung
x / @		Nullbyte/Nullfüller bis Position
X		Ein Byte zurück

Jedem Zeichen darf dabei eine Dezimalzahl folgen, die die Zahl der Wiederholungen angibt; der Stern steht dabei für alle noch folgenden Zeichen. Wenn dem Format %n vorangeht, gibt `unpack` statt dessen eine n-bit-Prüfsumme. Leerzeichen dürfen verwendet werden, um die Lesbarkeit zu erhöhen.

Stringfunktionen

`chomp` *Liste*†
: Entfernt die Zeilenenden von allen Elementen der Liste, gibt die Anzahl der entfernten Zeichen zurück.

`chop` *Liste*†
: Entfernt das letzte Zeichen von allen Elementen der Liste, gibt das letzte entfernte Zeichen zurück.

`crypt` *Text, Salt*
: Verschlüsselt einen Text.

`eval` *Ausdr*†
: *Ausdr* wird geparst und als Perl-Programm ausgeführt. Der zurückgegebene Wert ist der des letzten ausgewerteten Ausdrucks. Wenn ein Syntax- oder Laufzeitfehler auftritt, gibt `eval` einen undefinierten String zurück, und $@ wird auf die Fehlermeldung gesetzt. Siehe auch unter eval im Abschnitt „Diverses".

`index` *Str, Substr* [, *Offset*]
: Gibt die Position von *Substr* in *Str* (ggf. erst ab Offset) zurück. Wenn der Teilstring nicht gefunden wird, wird -1 zurückgegeben (aber siehe auch $[unter „Spezielle Variablen").

`length` *Ausdr*†
: Gibt die Anzahl von Zeichen in *Ausdr* zurück.

`lc` *Ausdr*
: Gibt *Ausdr* kleingeschrieben zurück.

`lcfirst` *Ausdr*
: Gibt *Ausdr* mit kleingeschriebenem Anfangsbuchstaben zurück.

`quotemeta` *Ausdr*
: Gibt *Ausdr* zurück, wobei alle Metazeichen für reguläre Ausdrücke gequotet sind.

`rindex` *Str*, *Substr* [, *Offset*]
: Gibt die Position des letzten Auftretens von *Str* rechts von *Offset* zurück.

`substr` *Ausdr*, *Offset* [, *Länge*]
: Extrahiert einen String der angegebenen Länge an der Stelle *Offset* aus *Ausdr* und gibt ihn zurück. Wenn *Offset* negativ ist, wird vom Ende des Strings gezählt. Kann auch einen Wert zugewiesen bekommen.

`uc` *Ausdr*
: Gibt *Ausdr* großgeschrieben zurück.

`ucfirst` *Ausdr*
: Gibt *Ausdr* mit großgeschriebenem Anfangsbuchstaben zurück.

Perl

Feld- und Listenfunktionen

`delete` $*Hash*{*schlüssel*}
: Löscht den angegebenen Wert aus der angegebenen Hash-Tabelle. Gibt den gelöschten Wert zurück (sofern *hash* nicht an ein Paket gebunden wurde, das dies nicht unterstützt).

`each` %*Hash*
: Gibt ein Feld aus zwei Elementen zurück, das aus dem Hash-Schlüssel und dem Wert des nächsten Elements der Hash-Tabelle besteht. Die Einträge werden in scheinbar zufälliger Reihenfolge zurückgegeben. Wenn alle Elemente des Feldes übergeben wurden, wird ein Nullfeld zurückgegeben. Beim nächsten Aufruf von `each` beginnt das Auslesen dann von vorne.

`exists` *Ausdr*†
: Überprüft, ob der angegebene Hash-Schlüssel in seinem Hash-Feld existiert.

`grep` *Ausdr*, *Liste*

`grep` *Block Liste*
: Evaluiert *Ausdr* oder *Block* für jedes Element der *Liste*, wobei $_ lokal auf das Element zeigt. Änderungen an $_ ändern das zugehörige Element der *Liste*. Gibt ein Feld aus den Elementen der *Liste* zurück, für die *Ausdr* zu true evaluierte.

`join` *Ausdr*, *Liste*
: Fügt die Einzelstrings der *Liste* zu einem einzigen String zusammen, wobei die Felder durch den Wert von *Ausdr* getrennt sind, und gibt diesen zurück.

`keys` *%Hash*
: Gibt ein Feld aller Schlüssel des angegebenen Hashs zurück.

`map` *Ausdr, Liste*

`map` *Block Liste*
: Evaluiert *Ausdr* oder *Block* für jedes Element der *Liste*, wobei $_ lokal auf das Element zeigt. Änderungen an $_ wirken sich auch auf das zugehörige Element der *Liste* aus. Gibt die Liste der Ergebnisse zurück.

`pop` *@Feld*
: Entfernt den letzten Wert des Feldes und gibt ihn zurück.

`push` *@Feld, Liste*
: Legt die Werte von *Liste* am Ende des Feldes ab.

`reverse` *Liste*
: Gibt im Feldkontext die Elemente von *Liste* in umgekehrter Reihenfolge zurück. Gibt im skalaren Kontext das erste Element der *Liste* mit umgekehrter Bytefolge zurück.

`scalar` *@Feld*
: Gibt die Anzahl von Elementen des Feldes zurück.

`scalar` *%Hash*
: Gibt den Wert `true` zurück, wenn das Hash definierte Elemente enthält.

`shift` [*@Feld*]
: Schiebt den ersten Wert des Feldes heraus und gibt ihn zurück, wobei sich alle anderen Elemente um eins nach unten bewegen. Wenn *@Feld* fehlt, wird in der Hauptroutine @ARGV und in den Unterroutinen @_ verschoben.

`sort` [*Subroutine*] *Liste*
: Sortiert die Liste und gibt den sortierten Wert des Feldes zurück. *Subroutine* muß, wenn angegeben, Werte von kleiner 0, 0 oder größer 0 zurückgeben, wenn die Elemente voreinander, nebeneinander oder hintereinander einsortiert werden sollen (sie stehen der Unterroutine als $a und $b zur Verfügung). *Subroutine* kann der Name einer benutzerdefinierten Routine oder ein Block sein.

`splice` *@Feld, Offset* [, *Länge* [, *Liste*]]
: Entfernt die Elemente von @Feld, die durch *Offset* und *Länge* definiert werden, und ersetzt sie durch *Liste* (sofern angegeben). Gibt die entfernten Elemente zurück.

`split` [*Muster* [, *Ausdr*† [, *Limit*]]]
: Teilt einen String in ein Feld von Strings und gibt dieses zurück. Wenn *Limit* angegeben ist, werden maximal so viele Felder erzeugt. Wenn *Muster* fehlt, trennt split an den vordefinierten Leerräumen (*Whitespace*). Steht `split` nicht im Arraykontext, gibt es die Anzahl von Feldern zurück und splittet nach @_. Siehe auch unter „Suche- und Ersetze-Funktionen“.

`unshift` *@Feld, Liste*
: Stellt *Liste* an den Beginn von *Feld* und gibt die Anzahl von Elementen zurück, die nun im Feld sind.

`values` *%Hash*
: Gibt ein Feld zurück, das aus allen Werten in dem angegebenen Hash besteht.

Reguläre Ausdrücke

Jedes Zeichen paßt auf sich selbst, sofern es keines der besonderen Zeichen + ? . * ^ $ ( ) [ ] { } | \ ist. Die besondere Bedeutung dieser Zeichen kann durch ein vorangestelltes \ umgangen werden.

. Paßt auf ein beliebiges Zeichen, jedoch nicht auf ein Newline, solange es sich nicht um einen einzeiligen Vergleich handelt (siehe `m//s`).

(...) Fügt eine Folge von Musterelementen zu einem Element zusammen.

^ Paßt auf den Anfang. Im Mehrzeilenmodus auch auf jeden Zeilenanfang (siehe auch m//m).

$ Paßt auf das Zeilenende. Im Mehrzeilenmodus auch vor jedem Newline.

[...]
Bezeichnet eine Gruppe von passenden Zeichen. [^...] negiert die Gruppe von Zeichen.

(... | ... | ...)
Paßt auf eine beliebige Alternative.

(?# *Text*)
Kommentar

(?: *Regexp*)
Entspricht (*Regexp*), jedoch ohne Rückwärtsreferenz.

(?= *Regexp*)
Positive Look-Ahead-Bedingung.

(?! *Regexp*)
Negative Look-Ahead-Bedingung.

(? *Modifikator*)
Eingefügter Modifikator für das Pattern-Matching. Modifikator kann dabei eines oder mehrere der Zeichen `i`, `m`, `s` oder `x` sein.

Quantifizierte Teilmuster passen sooft wie möglich. Wenn ihnen ein ? folgt, passen sie so selten wie möglich. Dies sind die Quantifikatoren:

+ Läßt das vorangehende Musterelement einmal oder mehrmals passen.

? Paßt kein- oder einmal.

* Paßt keinmal bis mehrmals.

{*n*,*m*} Gibt die minimale (*n*) und maximale (*m*) Anzahl der Übereinstimmungen an. {*n*} bedeutet, daß exakt *n* Übereinstimmungen vorliegen müssen, {*n*,} bedeutet mindestens *n* Übereinstimmungen.

Ein \ übergeht jede spezielle Funktion des folgenden Zeichens, wenn es nicht alphanumerisch ist, aber es verleiht den meisten normalen Zeichen eine besondere Bedeutung:

\w Paßt auf alphanumerische Zeichen inklusive _

\W	Paßt auf nichtalphanumerische Zeichen.
\s	Paßt auf Whitespace, \S paßt auf alles andere.
\d	Paßt auf numerische Zeichen, \D auf alle anderen.
\A	Paßt auf den Anfang eines Strings, \Z auf das Ende.
\b	Paßt auf Wortgrenzen, \B auf alle anderen Stellen.
\G	Paßt dort, wo die vorherige Suche mit `m//g` endete.
\n, \r, \f, \t, ...	Haben die übliche Bedeutung.
\w, \s und \d	Können auch in Zeichenklassen verwendet werden. \b bezeichnet in diesem Kontext einen Backspace.

Rückwärtsreferenzen:

\1 ... \9	Bezeichnen Teilübereinstimmungen innerhalb der Übereinstimmung, angeordnet in ( ).
\10 und höher	Kann ebenfalls verwendet werden, wenn das Muster entsprechend viele Teilausdrücke hat.

Siehe auch im Abschnitt „Spezielle Variablen" unter $1...$9, $+, $&, $` und $'.

Mit dem Modifikator `x` können Whitespaces in allen Mustern verwendet werden, um die Lesbarkeit zu erhöhen.

Suche- und Ersetze-Funktionen

[*Ausdr* =~] [`m`] /*Muster*/ [`g`] [`i`] [`m`] [`o`] [`s`] [`x`]

Durchsucht *Ausdr* (Default: $_) nach einem Muster. Wenn Sie ein `m` voranstellen, können Sie auch fast jedes andere Trennzeichenpaar verwenden. Wenn im Feldkontext verwendet, wird ein Feld zurückgeliefert, das aus den Teilausdrücken besteht, die von den Klammern im Muster bezeichnet werden, z.B. ($1, $2, $3, ...).

Optionale Modifikatoren: `g` erlaubt so viele Übereinstimmungen wie möglich, `i` sucht ohne Berücksichtigung von Groß/Kleinschreibung, `o` interpoliert Variablen nur einmal, `m` behandelt den String wie mehrere Zeilen, `x` erlaubt die Erweiterungen regulärer Ausdrücke.

Wenn *Muster* leer ist, wird das letzte Muster von einer Suche- oder Ersetze-Operation verwendet.

Mit `g` kann die Suche auch als Iterator in skalarem Kontext verwendet werden.

?*Muster*?

Dies entspricht der normalen Suche mit /*Muster*/, abgesehen davon, daß es zwischen zwei Aufrufen des `reset`-Operators nur einmal paßt.

[$*Var* =~] `s`/*Muster*/*Ersatz*/ [`e`] [`g`] [`i`] [`m`] [`o`] [`s`] [`x`]

Durchsucht einen String nach einem Muster und ersetzt es, sobald es gefunden ist, durch den Ersatztext. Zurückgegeben wird die Anzahl der vorgenommenen Ersetzungen, ansonsten `false`.

Optionale Modifikatoren: `g` ersetzt alle Vorkommen des Musters, `e` evaluiert den Ersatzstring als Perl-Ausdruck, die anderen Modifikatoren entsprechen denen der Suche-Operation. Statt der Schrägstriche kann auch fast jedes andere Trennzeichen verwendet werden. Bei Verwendung einfacher Anführungszeichen werden die Strings innerhalb der Trenner nicht interpoliert, ansonsten erfolgt eine normale Interpolation wie zwischen doppelten Anführungszeichen.

Wenn klammernde Trenner verwendet werden, können *Muster* und *Ersatz* jeweils eigene Trenner haben, z.B. s(foo)[bar]. Wenn *Muster* leer ist, wird das letzte Muster aus einer Suche- oder Ersetze-Operation verwendet.

[$*Var* =~] `tr/`*Suche-Liste*/*Ersetze-Liste*/ [c] [d] [s]

Übersetzt alle Vorkommen eines Zeichens in der *Suche-Liste* mit dem entsprechenden Zeichen der *Ersetze-Liste*. Es gibt die Anzahl der ersetzten Zeichen zurück. `tr` kann auch durch `y` ersetzt werden.

Optionale Modifikatoren: `c` komplementiert die *Suche-Liste*, `d` löscht alle Zeichen in der *Suche-Liste*, die kein entsprechendes Zeichen in der *Ersetze-Liste* haben, `s` ersetzt alle Zeichenfolgen, die durch das gleiche Zeichen ersetzt werden, mit nur einem einzigen Zeichen.

`pos` *Skalar*

Gibt die Position, an der die letzte Suche mit `m//g` endete, für *Skalar* zurück. Kann auch einen Wert zugewiesen bekommen.

`study` [$*Var*†]

Führt eine vorbereitende Untersuchung der Variable $*Var* in bezug auf zahlreiche anstehende Mustersuchen in ihrem Inhalt durch, bevor die Variable das nächste Mal verändert wird.

Dateitestoperatoren

Diese unären Operatoren haben nur ein Argument, entweder den Namen oder das Handle einer Datei. Sie testen die entsprechende Datei auf Vorhandensein der getesteten Eigenschaft. Wenn das Argument fehlt, wird $_ getestet (außer bei -t, das die Standardeingabe prüft). Wenn das spezielle Argument _ (Unterstrich) übergeben wird, werden die Daten des letzten Tests oder des letzten Aufrufs von `stat` verwendet.

-r -w -x

Datei ist lesbar/schreibbar/ausführbar unter der effektiven UID/GID.

-R -W -X

Datei ist lesbar/schreibbar/ausführbar unter der realen UID/GID.

-o -O

Datei gehört der effektiven/realen UID.

-e -z

Datei existiert/hat Größe Null.

-s Datei existiert und ist länger als 0 Bytes. Gibt Größe zurück.

-f -d
: Datei ist normales Datei/Verzeichnis.

-l -S -p
: Datei ist ein symbolischer Link/ein Socket/eine mit (einem Datei-)Namen versehene Pipe (FIFO).

-b -c
: Datei ist Block-/Character-Devicedatei.

-u -g -k
: Datei hat setuid/setgid/sticky-Bit gesetzt.

-t Testet, ob das Dateihandle (Default: Standardeingabe) auf ein tty zeigt.

-T -B
: Datei ist eine Textdatei/Binärdatei. -T und -B geben beide bei einer leeren Datei oder am Dateiende, wenn ein Dateihandle getestet wird, `true` zurück.

-M -A -C
: Testet Datum der letzten Änderung/des letzten Zugriffs/der Inode, gemessen in Tagen. Der übergebene Wert gibt das Dateialter zur Startzeit des Skripts zurück. Siehe auch unter $^T im Abschnitt „Spezielle Variablen".

Dateioperationen

Die Funktionen, die mit einer Liste von Dateien arbeiten, geben die Anzahl der Dateien zurück, bei denen sie erfolgreich waren.

`chmod` *Liste*
: Ändert die Zugriffsrechte für eine Liste von Dateien. Das erste Element der Liste muß die numerische Modusangabe sein.

`chown` *Liste*
: Ändert Eigentümer und Gruppe für eine Liste von Dateien. Die ersten beiden Elemente der Liste müssen die numerischen UID und GID sein.

`truncate` *Datei, Größe*
: Kürzt die *Datei* auf die angegebene *Größe. Datei* kann entweder der Name oder das Handle einer Datei sein.

`link` *AlteDatei, NeueDatei*
: Erzeugt einen Link auf die alte Datei unter dem neuen Namen.

`lstat` *Datei*
: Entspricht `stat`, folgt einem abschließenden symbolischen Link jedoch nicht.

`mkdir` *Dir, Modus*
: Erzeugt ein Verzeichnis mit den angegebenen Zugriffsrechten. Bei einem Fehler wird $! gesetzt.

`readlink` *Ausdr*†
: Gibt den Wert eines symbolischen Links zurück.

`rename` *AlterName, Neuer*
Name Ändert den Namen einer Datei.

`rmdir` *Dateiname*†
Löscht das Verzeichnis, wenn es leer ist. Setzt bei Fehler $!.

`stat` *Datei*
Übergibt ein Feld mit 13 Elementen (0:$dev, 1:$ino, 2:$mode, 3:$nlink, 4:$uid, 5:$gid, 6:$rdev, 7:$size, 8:$atime, 9:$mtime, 10:$ctime, 11:$blksize, 12:$blocks). *Datei* kann dabei ein Handle, ein zu einem Dateinamen evaluierender Ausdruck oder der Platzhalter _ sein, der durch die letzte bei einem Test oder einem `stat`-Aufruf verwendete Datei ersetzt wird. Wenn `stat` fehlschlägt, wird eine leere Liste zurückgegeben.

`symlink` *AlteDatei, NeueDatei*
Legt einen neuen Namen der Datei als symbolischen Link auf den alten an.

`unlink` *Liste*
Löscht eine Liste von Dateien.

`utime` *Liste*
Ändert Zugriffs- und Änderungszeiten. Die ersten zwei Elemente der Liste müssen numerische Werte für den letzten Zugriff und die letzte Änderung sein.

Ein-/Ausgabe

In Ein-/Ausgabe-Operationen darf *Dateihandle* entweder ein Handle sein, wie es von der open-Operation zurückgeliefert wird, ein vordefiniertes Handle (z.B. STDOUT) oder eine skalare Variable, die zum Namen des zu verwendenden Dateihandles evaluiert.

<*Dateihandle*>
Liest in skalarem Kontext eine einzelne Zeile aus der unter Dateihandle geöffneten Datei. Im Feldkontext wird die gesamte Datei gelesen.

<>
Liest aus der Eingabe, die aus den in @ARGV angegebenen Dateien besteht, oder der Standardeingabe, wenn keine Argumente übergeben wurden.

`binmode` *Dateihandle*
Stellt sicher, daß die unter *Dateihandle* geöffnete Datei im Binär- anstatt im Textmodus bearbeitet wird (unter UNIX eine Null-Operation).

`close` *Dateihandle*
Schließt die mit dem Dateihandle assoziierte Datei oder Pipe.

`dbmclose` *%Hash*
Überholt, verwenden Sie statt dessen `untie`.

`dbmopen` *%Hash, DbmName, Modus*
Überholt, verwenden Sie statt dessen `tie`.

`eof` *Dateihandle*
Gibt true zurück, wenn die nächste Leseoperation EOF liefern würde oder wenn die Datei nicht geöffnet ist.

`eof`
: Gibt den EOF-Status für die letzte gelesene Datei zurück.

`eof()`
: Gibt EOF für die aus den Dateien in der Befehlszeile geformte Pseudodatei an.

`fcntl` *Dateihandle, Funktion, $Var*
: Implementiert fcntl(2). Die Funktion hat vom Standard abweichende Rückgabewerte.

`fileno` *Dateihandle*
: Gibt den Dateideskriptor für die angegebene (offene) Datei zurück.

`flock` *Dateihandle, Operation*
: Ruft flock(2) für die Datei auf. Operation wird durch Addition von 1 (shared), 2 (exclusive), 4 (non-blocking) und 8 (unlock) gebildet.

`getc` [*Dateihandle*]
: Liefert das nächste Zeichen der Datei oder einen leeren String, falls die Datei zu Ende ist. Wenn *Dateihandle* fehlt, wird von der Standardeingabe gelesen.

`ioctl` *Dateihandle, Funktion, $Var*
: Implementiert *ioctl*(2). Die Funktion hat vom Standard abweichende Rückgabewerte.

`open` *Dateihandle* [, *Dateiname*]
: Öffnet eine Datei und ordnet ihr *Dateihandle* zu. Wenn der *Dateiname* fehlt, muß die skalare Variable, die denselben Namen wie das *Dateihandle* hat, den Dateinamen enthalten.

Die folgenden Konventionen für den Dateinamen gelten beim Öffnen einer Datei:

"*Datei*"
: Öffnet *Datei* für die Eingabe. Auch "<*Datei*".

">*Datei*"
: Öffnet *Datei* für die Ausgabe, wobei es sie anlegt, wenn nötig.

">>*Datei*"
: Öffnet die *Datei* im Anfügemodus.

"+>*Datei*"
: Öffnet die *Datei* zum Lesen und Schreiben.

"| *Kommando*"
: Öffnet eine Pipe zum *Kommando*, führt einen Fork aus, wenn *Kommando* ein – ist.

"*Kommando*" |
: Öffnet eine Pipe vom *Kommando*, führt einen Fork aus, wenn *Kommando* ein – ist.

Datei darf auch &*Dateihandle* sein, in diesem Fall wird das neue Dateihandle mit dem (zuvor geöffneten) Handle *Dateihandle* verknüpft. Wenn &=*n* ist, wird Datei mit dem angegebenen Dateideskriptor verknüpft. `open` liefert bei Versagen `undef` zurück, anderenfalls `true`.

`pipe` *Lesehandle, Schreibhandle*
: Liefert ein Paar verbundener Pipes zurück.

`print` [*Dateihandle*] [*Liste*†]
: Gibt die Elemente von *Liste* aus, wobei sie nach Bedarf in Strings verwandelt werden.

Wenn *Dateihandle* fehlt, wird per Voreinstellung auf die Standardausgabe ausgegeben (oder in den letzten angewählten Ausgabekanal, siehe `select`).

`printf` [*Dateihandle*] [*Liste*]
: Äquivalent zu `print` *Dateihandle* und `sprintf` *Liste*.

`read` *Dateihandle*, $*Var*, *Anzahl* [,*Offset*]
: Liest der *Anzahl* entsprechend binäre Bytes aus der Datei in die *Variable* (ggf. an Position *Offset*). Gibt die Anzahl der tatsächlich gelesenen Bytes zurück.

`seek` *Dateihandle*, *Position*, *Whence*
: Wählt eine beliebige *Position* in der Datei aus. Gibt bei Erfolg `true` zurück.

`select` [*Dateihandle*]
: Gibt das derzeit ausgewählte Dateihandle zurück. Setzt das Defaulthandle für Ausgabeoperationen, wenn *Dateihandle* angegeben ist.

`select` *rbits,wbits,nbits,timeout*
: Führt den Systemaufruf *select*(2) mit diesen Parametern aus.

`sprintf` *Format*, *Liste*
: Gibt einen String zurück, der nach den üblichen *printf*(3)-Konventionen formatiert wurde.

`sysread` *Dateihandle*, $*Var*, *Anzahl* [, *Offset*]
: Liest *Anzahl* Bytes an der Stelle *Offset* in die Variable $*Var*.

`syswrite` *Dateihandle*, *Skalar*, *Anzahl* [, *Offset*]
: Schreibt *Anzahl* Bytes aus dem *Skalar* (ggf. ab *Offset*) in die Datei.

`tell` [*Dateihandle*]
: Gibt die aktuelle Position in der Datei zurück. Wenn *Dateihandle* fehlt, wird die zuletzt gelesene Datei angenommen.

Formate

`formline` *Schablone*, *Liste*
: Formatiert die *Liste* gemäß der *Schablone* und fügt das Ergebnis in $^A zusammen.

`write` [*Dateihandle*]
: Schreibt einen formatierten Datensatz in die angegebene Datei, wobei das mit dieser Datei assoziierte Format verwendet wird.

Formate werden wie folgt definiert:

`format` [*Name*] =
formatliste
.

formatliste gibt Zeilenschablonen an und enthält die Argumente, die den Feldern in den Zeilen ihre Werte geben. *Name* ist auf die Standardausgabe voreingestellt.

Perl

Schablonenfelder sind:

`@<<<...`	Linksbündiges Feld, wiederholen Sie <, um die gewünschte Länge anzugeben.
`@>>>...`	Rechtsbündiges Feld
`@|||...`	Zentriertes Feld
`@#.##...`	Numerisches Format mit vorgegebenem Dezimalpunkt
`@*`	Ein mehrzeiliges Feld

Verwenden Sie ^ statt @ für mehrzeiligen Blocksatz.

Verwenden Sie ~ am Anfang einer Zeile, um unerwünschte Leerzeilen zu vermeiden.

Verwenden Sie ~~ am Anfang einer Zeile, um diese Formatzeile zu wiederholen, bis alle Felder verwertet sind.

Setzen Sie $- auf Null, um beim nächsten `write` einen Zeilenvorschub auszulösen.

Siehe auch $^, $~, $^A, $^F, $– und $= im Abschnitt „Spezielle Variablen“.

Funktionen zum Lesen von Verzeichnissen

`closedir` *DirHandle*
: Schließt ein von `opendir` geöffnetes Verzeichnis.

`opendir` *DirHandle, DirName*
: Öffnet ein Verzeichnis unter dem angegebenen Handle.

`readdir` *DirHandle*
: Gibt den nächsten Eintrag (oder ein Feld von Einträgen) aus dem angebenen Verzeichnis zurück.

`rewinddir` *DirHandle*
: Positioniert das Verzeichnis auf den Anfang.

`seekdir` *DirHandle, Pos*
: Setzt die Position für ein readdir auf dieses Verzeichnis.

`telldir` *DirHandle*
: Gibt die aktuelle Position im Verzeichnis zurück.

Interaktion mit dem System

`alarm` *Ausdr*
: Sieht einen SIGALRM vor, der nach *Ausdr* Sekunden ausgelöst wird.

`chdir` [*Ausdr*]
: Wechselt das aktuelle Arbeitsverzeichnis. Wenn *Ausdr* fehlt, wird $ENV{"HOME"} oder $ENV{"LOGNAME"} verwendet.

`chroot` *Dateiname*†
: Wechselt das Rootverzeichnis für den Prozeß und die Kindprozesse.

`die` [*Liste*]
: Gibt den Wert der Liste auf die Standardfehlerausgabe aus und beendet das Programm mit dem aktuellen Wert von $! (errno). Wenn $! 0 ist, endet das Programm mit ($? >> 8). Ist auch das 0, endet das Programm mit dem Exitcode 255. Default für *Liste* ist „Died".

`exec` *Liste*
: Führt das Systemkommando in *Liste* aus, kehrt nicht zurück.

`exit` [*Ausdr*]
: Kehrt unmittelbar zum System zurück mit dem Wert aus *Ausdr* bzw. dem Default 0. Ruft END-Routinen und Objektdestruktoren auf, bevor das Programm verlassen wird.

`fork`
: Führt die Systemroutine *fork*(2) aus. Gibt die Prozeß-ID des Kindprozesses an den Elternprozeß zurück und 0 an den Kindprozeß.

`getlogin`
: Gibt den aktuellen Loginnamen zurück, wie er dem System bekannt ist.

`getprgp` [*pid*]
: Gibt die Prozeßgruppe für den Prozeß *pid* zurück. (Ist *pid* 0 oder wird keine Angabe gemacht, ist der aktuelle Prozeß gemeint.)

`getppid`
: Gibt die Prozeß-ID für den Elternprozeß zurück.

`getpriority` *which*, *who*
: Gibt die Priorität für einen Prozeß, eine Prozeßgruppe oder einen Anwender zurück.

`glob` *Muster*
: Gibt eine Liste von Dateinamen zurück, die auf das Shell-Muster *Muster* passen.

`kill` *Liste*
: Sendet ein Signal an eine Liste von Prozessen. Das erste Element der Liste muß das zu sendende Signal sein (entweder numerisch oder sein Name als String).

`setpgrp` *pid*, *pgrp*
: Setzt die Prozeßgruppe für den Prozeß *pid* (0 bezeichnet den aktuellen Prozeß).

`setpriority` *which*, *who*, *Priorität*
: Setzt die aktuelle Priorität für einen Prozeß, eine Prozeßgruppe oder einen Anwender.

`sleep` [*Ausdr*]
: Läßt das Programm für *Ausdr* Sekunden schlafen oder bei fehlendem *Ausdr* für immer. Gibt die Zahl von Sekunden zurück, die das Programm tatsächlich schlafend verbracht hat.

`syscall` *Liste*
: Ruft die Systemfunktion auf, die erstes Element der Liste ist, wobei der Rest der Liste in Form von Argumenten übergeben wird.

`system` *Liste*
: Entspricht exakt `exec` *Liste*, abgesehen davon, daß vorher ein Fork ausgeführt wird und der Elternprozeß schläft, bis der Kindprozeß abgeschlossen ist.

`times`
: Gibt ein Feld aus vier Elementen zurück (0:$user, 1:$system, 2:$cuser,3:$csystem), das die User- und Systemzeit für den Prozeß und die Kinder des Prozesses in Sekunden angibt.

`umask` [*Ausdr*]
: Setzt die umask für den Prozeß und liefert die alte zurück. Wenn *Ausdr* fehlt, wird der aktuelle umask-Wert zurückgegeben.

`wait`
: Wartet auf die Beendigung eines Kindprozesses und liefert die Prozeß-ID des beendeten Prozesses zurück (-1, falls keiner existiert). Der Status wird in $? zurückgegeben.

`waitpid` *pid, Flags*
: Führt die gleiche Funktion wie der entsprechende Systemaufruf aus.

`warn` [*Liste*]
: Gibt wie `die` die Nachricht auf die Standardfehlerausgabe aus, beendet den Prozeß jedoch nicht. Voreinstellung ist „Warning: Something's wrong".

Netzwerkfunktionen

`accept` *Neuer_Socket, Allgemeiner_Socket*
: Akzeptiert einen neuen Socket.

`bind` *Socket, Name*
: Bindet den *Namen* an den *Socket.*

`connect` *Socket, Name*
: Verbindet den *Namen* mit dem *Socket.*

`getpeername` *Socket*
: Gibt die Socketadresse am anderen Ende des Sockets zurück.

`getsockname` *Socket*
: Gibt den Namen des Sockets zurück.

`getsockopt` *Socket, Level, Optname*
: Gibt die Socketoptionen zurück.

`listen` *Socket, Queuegröße*
: Geht am angegebenen *Socket* „auf Empfang".

`recv` *Socket, Skalar, Länge, Flags*
: Empfängt auf *Socket* eine Nachricht.

`send` *Socket, Nachricht, Flags* [, *An*]
: Sendet an den *Socket* eine Nachricht.

`setsockopt` *Socket, Level, Optname, Optval*
: Setzt die entsprechende Socket-Option.

`shutdown` *Socket, How*
: Schließt einen *Socket.*

`socket` *Socket, Domain, Typ, Protokoll*
: Erzeugt einen *Socket* in der *Domain* mit vorgegebenem *Typ* und *Protokoll.*

`socketpair` *Socket1, Socket2, Domain, Typ, Protokoll*
: Entspricht `socket`, erzeugt jedoch ein Paar bidirektionaler Sockets.

System V IPC

Sie müssen `require "sys/ipc.ph"` verwenden, bevor Sie die symbolischen Namen dieser Operationen benutzen können.

`msgctl` *id, Kommando, Argumente*
: Ruft *msgctl*(2) auf. Wenn *Kommando* &IPC_STAT ist, muß *Argumente* eine Variable sein.

`msgget` *Schlüssel, Flags*
: Erzeugt eine Message-Queue für *Schlüssel.* Gibt den Identifikator der Queue zurück.

`msgsnd` *id, Nachricht, Flags*
: Sendet *Nachricht* an die Queue *id.*

`msgrcv` *id, $Var, Größe, Typ, Flags*
: Empfängt eine Nachricht aus der Queue *id* in die Variable *Var.*

`semctl` *id, semnum, Kommando, Argument*
: Ruft *semctl*(2) auf. Wenn *Kommando* &IPC_STAT oder &GETALL ist, muß *Argument* eine Variable sein.

`semget` *Schlüssel, nsems, Größe, Flags*
: Erzeugt für *Schlüssel* einen Satz von Semaphoren. Gibt den Identifikator zurück.

`semop` *Schlüssel, ...*
: Führt Semaphor-Operationen durch.

`shmctl` *id, Kommando, Argument*
: Ruft *shmctl*(2) auf. Wenn *Kommando* &IPC_STAT ist, muß *Argument* eine Variable sein.

`shmget` *Schlüssel, Größe, Flags*
: Erzeugt einen Speicherbereich für gemeinsamen Zugriff. Gibt den Identifikator für das Speichersegment zurück.

`shmread` id, *$Var, Position, Größe*
: Gibt maximal *Größe* Bytes aus dem Inhalt des gemeinsamen Speichersegments *id* ab der angegebenen *Position* in die Variable *Var* aus.

`shmwrite` *id, String, Position, Größe*
: Schreibt maximal *Größe* Bytes aus *String* in den Inhalt des gemeinsamen Speichersegments id an die angegebene *Position.*

Diverses

`defined` *Ausdr*
: Prüft, ob der Wert *Ausdr* tatsächlich einen Wert hat.

`do` *Dateiname*
: Führt *Dateiname* als Perl-Skript aus. Siehe auch unter `require` im Abschnitt „Unterroutinen, Pakete und Module".

`dump [` *Label* `]`
: Sofortiger Coredump. Läuft bei Reinkarnation bei *Label* weiter.

`eval {` *Ausdr*, ... `}`
: Führt den Code zwischen { und } aus. Laufzeitfehler werden wie bei `eval` (*Ausdr*) unter „Stringfunktionen" erläutert.

`local` *Var*

`local (` *Liste* `)`
: Legt den Gültigkeitsbereich der Variablen als lokal für den Block, die Unterroutine oder das eval-Statement fest.

`my` Var

`my (` *Liste* `)`
: Legt den lexikalischen Gültigkeitsbereich der Variablen als lokal für den Block, die Unterroutine oder das `eval`-Statement fest.

`ref` *Ausdr*†
: Gibt `true` zurück, wenn *Ausdr* eine Referenz ist. Gibt den Paketnamen zurück, wenn *Ausdr* mit `bless` einem Paket zugeordnet wurde.

`reset [` *Ausdr* `]`
: Setzt Suchen mit ?? zurück, so daß sie erneut arbeiten. *Ausdr* ist dabei eine Liste einzelner Buchstaben. Alle Variablen und Felder, die mit diesen Buchstaben beginnen, werden dabei auf ihren ursprünglichen Zustand zurückgesetzt. Betrifft ausschließlich das aktuelle Paket.

`scalar` *Ausdr*
: Erzwingt die Evaluation von *Ausdr* in skalarem Kontext.

`undef [` *lvalue* `]`
: Löscht die Definition von *lvalue*. Gibt immer den undefinierten Wert zurück.

`wantarray`
: Gibt `true` zurück, wenn der aktuelle Kontext einen Feldwert erwartet.

Information aus Systemdateien

`passwd`
Gibt ($name, $passwd, $uid, $gid, $quota, $comment, $gcos, $dir, $shell) zurück.

`endpwent`
: Beendet das Nachsehen in der passwd-Datei.

`getpwent`
: Holt die Informationen über den nächsten User.

`getpwnam` *name*
: Holt Informationen nach Namen.

`getpwuid` *uid*
: Holt Informationen nach User-ID.

`setpwent`
: Beginnt mit dem Nachsehen wieder von vorne.

`group`

Gibt ($name, $passwd, $gid, $members) zurück.

`endgrent`
: Beendet das Nachsehen in der group-Datei.

`getgrgid` *gid*
: Holt Informationen nach Group-ID.

`getgrnam` *name*
: Holt Informationen nach Gruppenname.

`getgrent`
: Holt Informationen über die nächste Gruppe.

`setgrent`
: Beginnt mit dem Nachsehen wieder von vorne.

`hosts`

Gibt ($name,$aliases,$addrtype,$length,@addrs) zurück.

`endhostent`
: Beendet das Nachsehen in der hosts-Datei.

`gethostbyaddr` *addr*, *addrtype*
: Holt Informationen nach IP-Adresse.

`gethostbyname` *name*
: Holt Informationen nach Hostnamen.

`gethostent`
: Holt Informationen über den nächsten Host.

`sethostent` *stayopen*
: Beginnt mit dem Nachsehen wieder von vorne.

`networks`

Gibt ($name, $aliases, $addrtype, $net) zurück.

`endnetent`
: Beendet das Nachsehen in der networks-Datei.

`getnetbyaddr` *addr*, *type*
: Holt Informationen nach Adresse und Typ.

`getnetbyname` *name*
: Holt Informationen nach Netzwerknamen.

`getnetent`
: Holt Informationen über das nächste Netz.

`setnetent` *stayopen*
: Beginnt mit dem Nachsehen wieder von vorne.

`services`
Gibt ($name, $aliases, $port, $proto) zurück.

`endservent`
: Beendet das Nachsehen in der services-Datei.

`getservbyname` *name, proto*
: Holt Informationen nach Name des Service.

`getservbyport` *port, proto*
: Holt Informationen nach Port des Service.

`getservent`
: Holt Informationen über den nächsten Service.

`setservent` *stayopen*
: Beginnt mit dem Nachsehen wieder von vorne.

`protocols`
Gibt ($name, $aliases, $proto) zurück.

`endprotoent`
: Beendet das Nachsehen in der protocols-Datei.

`getprotobyname` *Name*
: Holt Informationen nach Namen des Protokolls.

`getprotobynumber` *Nummer*
: Holt Informationen nach Nummer des Protokolls.

`getprotoent`
: Holt Informationen über das nächste Protokoll.

`setprotoent` *stayopen*
: Beginnt mit dem Nachsehen wieder von vorne.

Spezielle Variablen

Die folgenden Variablen sind global und sollten in Unterroutinen lokalisiert werden:

- `$_` Defaulteingabe und Suchraum für Mustersuche.
- `$.` Die aktuelle Eingabezeilennummer der zuletzt gelesenen Datei.
- `$/` Der Eingabeseparator, per Default Newline. Kann auch mehrere Zeichen beinhalten.
- `$,` Der Ausgabefeldseparator für den print-Operator.

$" Der Separator, der Feldelemente verbindet, die in einen String interpoliert wurden.

$\ Der Ausgabedatensatzseparator für den print-Operator.

$# Das Ausgabeformat für Zahlen. Überholt.

$* Auf 1 setzen, um Mehrzeilenvergleich für Strings zu aktivieren. Überholt, siehe auch die Modifikatoren m und s bei den „Suche- und Ersetze-Operationen".

$? Der vom letzten \`...\`-Kommando, Pipe-`close` oder `system`-Aufruf zurückgegebene Status.

$] Die Versionsnummer von Perl, z.B. 5.001.

$[Der Index des ersten Elementes eines Feldes und des ersten Zeichens in einem Teilstring. Default ist 0. Überholt.

$; Der Separator für die Emulation mehrdimensionaler Felder. Default ist „\034".

$! In numerischem Kontext der aktuelle Wert von errno. In Stringkontext der entsprechende Fehlertext.

$@ Die Perl-Fehlermeldung vom letzten `eval`- oder `do`-*Ausdr*-Kommando.

$: Die Menge der Zeichen, nach denen ein String getrennt werden darf, um Fortsetzungsfelder (mit ^ beginnend) in einem Format zu füllen.

$0 Der Name der Datei, die das aktuelle Perl-Skript beinhaltet. Kann einen Wert zugewiesen bekommen.

$$ Die Prozeß-ID des aktuell ausgeführten Perl-Programms. Wird durch `fork` (für den Kindprozeß) verändert.

$< Die reale UID des Prozesses.

$> Die effektive UID des Prozesses.

$(Die reale GID des Prozesses.

$) Die effektive GID des Prozesses.

$^A Der Akkumulator für `formline`- und `write`-Operationen.

$^D Die Debug-Flags, die Perl mit -D übergeben wurden.

$^F Der höchste Systemdateideskriptor, normalerweise 2.

$^I Edit-Erweiterung, wie sie Perl mit -i übergeben wurde.

$^L Seitenvorschubzeichen, wie es in Formaten verwendet wird.

$^P Internes Debugging-Flag.

$^T Die Zeit, zu der das Programm gestartet wurde (wie von `time` geliefert). Dieser Wert wird von den Dateitestoperatoren -M, -A und -C verwendet.

$^W Der Wert der -w-Option, wie er Perl übergeben wurde.

$^X Der Name, unter dem das gerade ausgeführte Programm gestartet wurde.

Die folgenden Variablen sind kontextabhängig und müssen nicht lokalisiert werden:

$% Die aktuelle Seitenzahl des jeweils selektierten Ausgabekanals.

$= Die Seitenlänge des aktuellen Ausgabekanals. Voreingestellt sind 60 Zeilen.

$- Die Anzahl Zeilen, die noch auf der Seite verbleibt.

$~ Der Name des aktuellen Report-Formats.

$^ Der Name des aktuellen Seitenanfang-Formats.

$| Sorgt nach jeder write- oder print-Operation für einen Flush auf den aktuellen Ausgabekanal, wenn ungleich 0.

$ARGV
: Der Name der aktuellen Datei, wenn von < > gelesen wird.

Die folgenden Variablen sind immer lokal zum jeweiligen Block:

$& Der String, für den bei der letzten Mustersuche Übereinstimmung festgestellt wurde.

$` Der String, der der letzten Übereinstimmung voranging.

$' Der String, der der letzten Übereinstimmung folgt.

$+ Die letzte Klammer, die mit dem letzten Suchmuster übereinstimmte.

$1...$9...
: enthalten die Teilmuster des zuletzt gefundenen Musters. $10 und höher sind nur verfügbar, wenn die Übereinstimmung entsprechend viele Teilmuster enthielt.

Spezielle Felder

@ARGV	Enthält die Befehlszeilenargumente für das Skript (ohne den Kommandonamen).
@EXPORT	Benennt die Methoden, die ein Paket per Default exportiert.
@EXPORT_OK	Benennt die Methoden, die ein Paket auf explizite Anforderungen exportieren kann.
@INC	Enthält eine Liste von Verzeichnissen, in denen gesucht wird, wenn ein Perl-Skript durch `do` *filename* oder `require`-Kommandos ausgeführt wird.
@ISA	Liste der Basisklassen eines Pakets.
@_	Parameterfeld für Unterroutinen. Wird auch von `split` verwendet, sofern nicht im Feldkontext.
%ENV	Enthält die aktuelle Umgebung.
%INC	Liste der Dateien, die mit require oder do eingefügt wurden.
%OVERLOAD	Kann verwendet werden, um Operatoren in einem Paket zu überladen.
%SIG	Wird verwendet, um Signalhandler für verschiedene Signale zu definieren.

Umgebungsvariablen

Perl verwendet die folgenden Umgebungsvariablen:

HOME
: Wird verwendet, wenn `chdir` kein Argument hat.

LOGDIR
: Wird verwendet, wenn `chdir` kein Argument hat und HOME nicht gesetzt ist.

PATH
: Wird bei der Ausführung von Subprozessen verwendet und beim Finden des Perl-Skripts, wenn -S verwendet wird.

PEL5LIB
: Eine durch Doppelpunkte getrennte Liste von Verzeichnissen, in denen Perl-Bibliotheken gesucht werden, bevor Perl in der Standard-Bibliothek und im aktuellen Verzeichnis nachsieht.

PERL5DB
: Das Kommando, um den Debugger-Code zu erhalten. Voreingestellt auf BEGIN { require 'perl5db.pl' }.

PERLLIB
: Wird statt PERL5LIB verwendet, wenn diese nicht definiert ist.

Der Perl-Debugger

Perl

Der symbolische Perl-Debugger wird mit perl -d aufgerufen.

h Gibt einen Hilfstext aus.

T Gibt eine Stackansicht aus.

s Einzelschritte.

n Einzelschritte um einen Unterroutineaufruf herum.

Eingabetaste
: Wiederholt letztes s oder n.

r kehrt aus der aktuellen Unterroutine zurück.

c [*Zeile*]
: Weiter (bis zur Zeile oder zum nächsten Breakpoint oder bis exit).

p *Ausdr*
: Gibt *Ausdr* aus.

l [*Bereich*]
: Gibt einen Bereich von Zeilen aus. *Bereich* kann eine Zahl sein, Start-Ende, Start+Anzahl oder Name einer Unterroutine. Wenn *Bereich* fehlt, wird das nächste Fenster aufgelistet.

w Gibt das Fenster um die aktuelle Programmzeile aus.

\- Gibt das vorherige Fenster erneut aus.

f *Datei*
: Wechselt in *Datei* und beginnt mit ihrer Auflistung.

l *Unterroutine*
: Listet die angegebene Unterroutine auf.

S Listet die Namen aller Unterroutinen auf.

/*Muster*/
: Sucht vorwärts nach *Muster.*

?*Muster*?
: Sucht rückwärts nach *Muster.*

b [*Zeile* [*Bedingung*]]
: Setzt einen Breakpoint bei *Zeile,* Default ist aktuelle Zeile.

b *Unterroutine* [*Bedingung*]
: Setzt einen Breakpoint auf die entsprechende Unterroutine.

d [*Zeile*]
: Löscht den Breakpoint an der angegebenen *Zeile.*

D Löscht alle Breakpoints.

L Listet Zeilen, die Breakpoints oder Aktionen definiert haben, auf.

a *Zeile Kommando*
: Setzt eine Aktion für eine Zeile.

A Löscht alle Aktionen.

< *Kommando*
: Setzt eine Aktion, die vor jedem Prompt des Debuggers ausgeführt wird.

> *Kommando*
: Setzt eine Aktion, die vor jedem s-, c- oder n-Kommando ausgeführt wird.

V [*Paket* [*Variable*]]
: Listet alle Variablen eines Pakets auf. Default ist main.

X [*Variable*]
: Wie V, jedoch für das aktuelle Paket.

! [[-]*Nummer*]
: Führt einen Befehl erneut aus. Voreingestellt ist der vorhergehende Befehl.

H [*-Anzahl*]
: Gibt die letzten *Anzahl* Befehle mit mehr als einem Buchstaben aus.

t Schaltet den Trace-Modus an und aus.

= [*Alias Wert*]
: Definiert einen Alias bzw. listet die aktuellen Aliases auf.

q Beendet den Debugger.

Befehl
: Führt *Befehl* als eine Perl-Anweisung aus.

KAPITEL 12

Überblick über die Befehle zum Programmieren

Dieses Kapitel enthält eine Reihe von Tabellen mit häufig benutzten Befehlen zur Programmierung.

Gebräuchliche Befehle

Im folgenden finden Sie eine Reihe von Tabellen von Befehlen, die häufig zur Software-Entwicklung verwendet werden. Diese und viele weitere Befehle werden detailliert im nächsten Abschnitt, „Alphabetische Befehlsübersicht", beschrieben.

Erstellen von Programmen

ar	Bibliotheksdateien erzeugen und aktualisieren.
as	Objektdateien erzeugen.
bison	Erzeugen von Parsing-Tabellen.
cpp	Präprozessor für C-Code.
g++	GNU C++ Compiler.
gcc	GNU C Compiler.
ld	Linker.
flex	Programm zur Generierung von lexikalischen Analyseprogrammen.
m4	Makroprozessor.
make	Werkzeug zur kontrollierten Erzeugung von Programmen.
rpcgen	Übersetzung von RPC-Spezifikationen in C-Programme.
yacc	Erzeugen von Parsing-Tabellen.

Verwaltung und Pflege von Programmen

ctags	Erzeugen von Symbollisten für den Editor **vi**.
etags	Erzeugen von Symbollisten für den Editor Emacs.
gdb	GNU Debugger.
gprof	Anzeige der Profile-Daten einer Objektdatei.

imake	CPP-Schnittstelle zu **make**.
make	Pflegen, Aktualisieren und neues Erzeugen von zusammengehörigen Programmen und Dateien.
nm	Ausgabe der Symboltabelle einer Objektdatei.
patch	Anbringen von Patches auf einer oder mehreren Dateien.
size	Ausgabe der Größe einer Objektdatei in Bytes.
strip	Entfernen der Symbole in einer Objektdatei.

Alphabetische Befehlsübersicht

ar

ar [-V] *Funktion* [*Argumente*] [*Name*] *Archiv* [*Dateien*]

Faßt eine Gruppe von *Dateien* zu einer Archivdatei *Archiv* zusammen. Dieser Befehl wird meistens dazu verwendet, Bibliotheksdateien zu erzeugen, die vom Linker **ld** weiter verwendet werden. Es darf nur ein *Funktions*-Buchstabe verwendet werden, an den aber zusätzliche *Argumente* angehängt werden dürfen (ohne Zwischenräume). *Name* bezeichnet den Namen einer Datei in *Archiv*. Wenn Dateien im Archiv ersetzt oder neue hinzugefügt werden, kann angegeben werden, daß sie vor oder hinter *Name* erscheinen sollen. Mit **-V** wird die Versionsnummer von **ar** auf Standardfehler ausgegeben.

*Funktions-**Buchstabe***

- d Löschen von *Dateien* aus *Archiv*.
- m Verschieben von *Dateien* nach *Archiv*.
- p Ausgeben von *Dateien* in *Archiv*.
- q Anhängen von *Dateien* an *Archiv*.
- r Ersetzen von *Dateien* in *Archiv*.
- t Anzeigen des Inhalts von *Archiv* oder von *Dateien*.
- x Löschen von *Dateien* aus *Archiv*.

Argumente

- a Wird bei **r** und **m** verwendet und gibt an, daß die Dateien hinter *Name* in das Archiv eingefügt werden sollen.
- b Wie **a**, aber es wird vor *Name* eingefügt.
- c Stillschweigendes Erzeugen des Archivs.
- i Wie **b**.
- l Nur zur Kompatibilität mit früheren Versionen, hat unter Linux keine Bedeutung.
- o Ursprüngliche Zeitstempel beibehalten.
- s Erzwingen, daß die Symboltabelle des Archivs neu erzeugt wird (nützlich nach Verwendung des **strip**-Befehls).
- u Wird im Zusammenhang mit **r** verwendet, um nur die Dateien zu ersetzen, die sich geändert haben, seit sie in das Archiv gestellt worden sind.
- v Verbose-Modus: zu jeder Aktion wird eine Beschreibung ausgegeben.

Beispiel

ar

Ersetzen von **mylib.a** durch alle Objektdateien des aktuellen Verzeichnisses:

```
ar r mylib.a `ls *.o`
```

as [*Optionen*] *Dateien*

as

Erzeugen einer Objektdatei aus allen in *Dateien* angegebenen Assembler-Quell-Dateien. Die Objektdateien erhalten den gleichen Namen wie die Quelldateien; das Suffix **.s** wird durch **.o** ersetzt. Zusätzlich zu den hier angegebenen Optionen gibt es auf manchen Systemen weitere, systemspezifische Optionen.

Optionen

-- [| *Dateien*]
: Einlesen der Quelldateien von der Standardeingabe oder, wenn das Pipe-Symbol angegeben ist, aus den *Dateien*.

-a[dhlns][=*Datei*]
: Wenn nur die **-a**-Option angegeben wird, werden Hochsprachen- und Assembler-Listing sowie die Symboltabelle ausgegeben. Die anderen Optionen schalten weitere Ausgaben ein oder aus:

-ad	Keine Debugging-Anweisungen ausgeben.
-ah	Wenn vorhanden, den Quelltext der Hochsprache mit ausgeben.
-al	Ausgeben eines Assembler-Listings.
-an	Unterdrücken der Verarbeitung von Ausdrücken.
-as	Ausgeben der Symbol-Tabelle.
=*Datei*	Die Ausgabe wird in die Datei *Datei* geschrieben.

-defsym *Symbol*=*Wert*
: Definiert das *Symbol* so, daß es den Wert *Wert* hat. *Wert* muß eine ganze Zahl sein.

-f Weglassen des Präprozessorlaufes.

-o *Objdatei*
: Schreiben der Ausgabe nach *Objdatei* (der Default ist *Datei*.o).

-v Ausgeben der Versionsnummer des Assemblers.

-I *Pfad*
: Durchsuchen des *Pfades* in **.include**-Anweisungen.

-K Ausgeben einer Warnung, bevor Differenzen (wie in .word SYM2-SYM1) verändert werden.

-L Lokale Symbole, die mit **L** beginnen, nicht entfernen.

-R Sowohl Daten- als auch Text-Segment in die text section schreiben.

-W „Quiet mode", unterdrücken von Ausgaben.

bison [*Optionen*] *Datei*

bison

Konvertiert eine Datei *Datei* mit einer kontext-freien Grammatik in Parsing-Tabellen, die in die Datei *Datei.c* geschrieben werden. **bison** ist weitgehend kompatibel mit **yacc** und auch danach benannt. Die Eingabedateien sollten

→

bison
←

das Suffix *.y* haben, die Ausgabedateien behalten den Basisnamen der Eingabedatei. Alle Lang-Optionen (also die, die mit -- anfangen) können statt dessen auch mit einem + eingeleitet werden.

Optionen

-b *Präfix*, --**file-prefix**=*Präfix*
: Alle Ausgabedateien fangen mit *Präfix* an.

-d, --**defines**
: Erzeugen einer Datei *Datei.h*, die #**define**-Anweisungen enthält. Diese bilden die von **bison** verwendeten Token-Codes auf die vom Benutzer bestimmten Token-Namen ab.

-r, --**raw**
: Verwenden von **bison**-Token-Nummern anstelle der **yacc**-kompatiblen Übersetzungen in *Datei.h*.

-k, --**token-table**
: Einfügen der Token-Namen und -Werte von YYNTOKENS, YYNNTS, YYNRULES und YYNSTATES in *Datei.c*.

-l, --**no-lines**
: Ausgabe von #**line**-Konstrukten in *Datei.c* unterdrücken. Sollte nur verwendet werden, wenn der Parser vollständig funktioniert.

-n, --**no-parser**
: Unterdrücken der Ausgabe des Parser-Codes; es werden nur die Deklarationen geschrieben. Alle Übersetzungen werden in den Body einer Switch-Anweisung geschrieben, die in die Datei *Datei.act* ausgegeben wird.

-o *Datei*, --**output-file**=*Datei*
: Schreiben der Ausgabe nach *Datei*.

-p *Präfix*, --**name-prefix**=*Präfix*
: Ersetzen von **yy** durch *Präfix* in allen extern verwendeten Symbolen.

-t, --**debug**
: Code zum Laufzeit-Debuggen mit einfügen.

-v, --**verbose**
: „Verbose-Mode". Ausgeben von Diagnose-Meldungen und Angaben zu den Parsing-Tabellen nach *Datei.output*.

-V, --**version**
: Ausgeben der Versionsnummer.

-y, --**yacc**, --**fixed-output-files**
: Verwenden der **yacc**-Konventionen bei der Benennung von Ausgabedateien.

---

c++

c++ [*Optionen*] *Dateien*

Siehe **g++**.

---

cc

cc [*Optionen*] *Dateien*

Siehe **gcc**.

cpp [*Optionen*] [*Eingabedatei* [*Ausgabedatei*]]

GNU-C-Präprozessor. **cpp** wird als erster Durchgang jedes C-Übersetzungsvorganges durch den **gcc**-Befehl aufgerufen. Die Ausgabe von **cpp** erfolgt in einer Form, die als Eingabe für den nächsten Durchgang des C Compilers verwendet werden kann. Normalerweise ruft **cpp** den **gcc** auf, sobald er seine Verarbeitung abgeschlossen hat. *Eingabedatei* und *Ausgabedatei* sind die Ein- und Ausgabedatei für den Präprozessor; die Defaults sind Standardeingabe und -ausgabe.

Optionen

-$ Zurückweisen von $ in Bezeichnern.

-dM Unterdrücken der normalen Ausgabe. Statt dessen wird eine Reihe von **#defines** geschrieben, welche die im Quelltext verwendeten Makros erzeugen.

-dD Ähnlich **-dM**. Vordefinierte Makros werden unterdrückt, und die Ergebnisse des Präprozessorlaufes werden mit ausgegeben.

-idirafter *Verz*
: Suchen von Header-Dateien in *Verz*, wenn die Header-Datei in keinem anderen Verzeichnis im Include-Pfad gefunden wurde.

-imacros *Datei*
: Verarbeitet zunächst die Makros in *Datei*, bevor die eigentliche Eingabedatei verarbeitet wird.

-include *Datei*
: Verarbeitet zunächst *Datei* vor der eigentlichen Eingabedatei.

-iprefix *Präfix*
: Wenn Verzeichnisse mit **-iwithprefix** hinzugefügt werden, wird diesen *Präfix* vorangestellt.

-iwithprefix *Verz*
: Anhängen von *Verz* an die Liste der Verzeichnisse, die durchsucht werden, wenn eine Header-Datei im Include-Pfad nicht gefunden werden konnte. Falls **-iprefix** angegeben wurde, wird dieser Präfix *Verz* vorangestellt.

-lang-c, -lang-c++, -lang-objc, -lang-objc++
: Angeben, daß der Quelltext in C, C++, Objective C oder Objective C++ ist.

-lint Alle lint-Befehle in Kommentaren werden als **#pragma lint** *Befehl* ausgegeben.

-nostdinc
: Nur die angegebenen, aber nicht die vorgegebenen Verzeichnisse werden nach Header-Dateien durchsucht.

-nostdinc++
: Unterdrücken des Durchsuchens von Verzeichnissen, die vermutlich C++-spezifische Header-Dateien enthalten.

-pedantic
: Ausgeben besonders vieler Warnungen.

-pedantic-errors
: Ausgeben eines fatalen Fehlers (der die Verarbeitung abbricht), wenn **-pedantic** eine Warnung ausgegeben hätte.

→

-traditional
Verhalten wie K&R-C, nicht ANSI-C.

-undef
Unterdrücken der Definition aller nicht-standardisierten Makros.

-C Ausgeben aller Kommentare (außer denen, die auf der **cpp**-Kommandozeile angegeben wurden). Defaultmäßig entfernt **cpp** alle C-Kommentare.

-D*Name*[=*Def*]
Definiert *Name* mit dem Wert *Def* wie mit **#define**. Wenn kein Wert =*Def* angegeben wird, erhält *Name* den Wert 1. **-D** hat eine niedrigere Priorität als **-U**.

-H Ausgeben der Pfadnamen eingefügter Dateien (einer pro Zeile) auf Standardfehler.

-I*Verz* Durchsuchen des Verzeichnisses *Verz* nach **#include**-Dateien, deren Namen nicht mit / beginnen, bevor die Standard-Verzeichnisse durchsucht werden. **#include**-Dateien, die in doppelten Anführungszeichen angegeben werden und deren Namen nicht mit / beginnen, werden zunächst im aktuellen Verzeichnis, dann in mit **-I** angegebenen Verzeichnissen und schließlich in Standard-Verzeichnissen gesucht.

-M [-MG]
Unterdrücken der normalen Ausgabe. Statt dessen werden **make**-Regeln ausgegeben, welche die Abhängigkeiten der Quelldatei beschreiben. Wenn **-MG** verwendet wird, wird angenommen, daß nicht vorhandene Header-Dateien während des Kompilationsprozesses im gleichen Verzeichnis wie die Quelldatei erzeugt werden.

-MD *Datei*
Ähnlich wie **-M**; die Ausgabe wird nach *Datei* geschrieben und die normale Verarbeitung nicht unterdrückt.

-MM Ähnlich wie **-M**; es werden nur Abhängigkeiten für Dateien ausgegeben, die mit **#include** "*Datei*" angegeben worden sind.

-MMD *Datei*
Ähnlich wie **-MMD**; es werden nur Abhängigkeiten von Header-Dateien des Benutzers ausgegeben.

-P Unterdrücken der Zeileninformationen, die im nächsten Durchgang vom Compiler verwendet werden würden.

-U*Name*
Entfernen jeder vorherigen Definition von *Name*. *Name* kann entweder ein internes, reserviertes Symbol des Präprozessors (auf Intel-Rechnern definiert **cpp unix** und **i386**) oder ein mit **-D** definiertes Symbol sein.

-Wcomment, -Wcomments
Ausgeben einer Warnung, wenn der Anfang eines verschachtelten Kommentars gefunden wird.

-Wtraditional
Ausgeben einer Warnung bei Konstrukten, die in ANSI-C anders als in K&R-C interpretiert werden.

Besondere Symbole

cpp kennt eine Reihe von besonderen Symbolen, u.a.:

__DATE__ Aktuelles Datum (z.B. May 27 1997)
__FILE__ Aktueller Dateiname (als C-String)
__LINE__ Aktuelle Zeilennummer (als Ganzzahl im Dezimalsystem)
__TIME__ Aktuelle Uhrzeit (z.B. 08:25:00)

Diese Symbole können überall, auch in Makros, verwendet werden, genau wie andere definierte Symbole auch. Die Zeilennummer und der Dateiname können mit der **#line**-Anweisung geändert werden.

Anweisungen

Alle Zeilen mit Anweisungen für **cpp** haben ein # in Spalte 1. Zwischen dem # und der Anweisung dürfen beliebig viele Leer- und Tabulatorzeichen stehen. Es gibt folgende Anweisungen:

#define *Name Token-String*
: Definiert ein Makro *Name* mit dem Wert *Token-String*. Weitere Vorkommen von *Name* werden durch *Token-String* ersetzt.

#define *Name(Arg, ..., Arg) Token-String*
: Makro-Substitution mit Argumenten. Bei einem Vorkommen von *Name* werden die dort angegebenen Argumente entsprechend in *Token-String* eingesetzt.

#undef *Name*
: Hebt die Definition von *Name* auf. Auf derselben Zeile dürfen keine weiteren Token folgen.

#ident *String*
: Fügt *String* in die comment section der Objektdatei ein.

#include "*Dateiname*", **#include**<*Dateiname*>
: Fügt den Inhalt von *Dateiname* an dieser Stelle in den Quelltext ein. Hinter dem abschließenden " oder > dürfen auf derselben Zeile keine weiteren Token mehr stehen.

#line *Integer-Konstante* "*Dateiname*"
: Gibt dem Compiler vor, daß die nächste Zeile als Zeile mit der Nummer *Integer-Konstante* zu betrachten ist. Falls *Dateiname* ebenfalls angegeben ist, wird dem Compiler weiterhin vorgegeben, daß die Eingabedatei als Datei mit dem Namen *Dateiname* zu betrachten ist.

#endif Abschluß eines Abschnitts von Programmzeilen, die mit **#if**, **#ifdef** oder **#ifndef** eingeleitet wurden. Dahinter sind keine weiteren Token auf derselben Zeile erlaubt.

#ifdef *Name*
: Alle folgenden Zeilen bis zum dazugehörigen **#endif** oder dem nächsten **#else** oder **#elif** werden nur in die Ausgabe übernommen, wenn *Name* definiert ist. Hinter *Name* sind auf derselben Zeile keine weiteren Token erlaubt.

#ifndef *Name*
: Alle folgenden Zeilen bis zum dazugehörigen **#endif** oder dem nächsten **#else** oder **#elif** werden nur in die Ausgabe

→

cpp
←

übernommen, wenn *Name* nicht definiert ist. Hinter *Name* sind auf derselben Zeile keine weiteren Token erlaubt.

#if *KonstanterAusdruck*
Alle folgenden Zeilen bis zum dazugehörigen **#endif** oder dem nächsten **#else** oder **#elif** werden nur in die Ausgabe übernommen, wenn *KonstanterAusdruck* zu einem von Null verschiedenen Wert ausgewertet wird. Hinter *KonstanterAusdruck* sind auf derselben Zeile keine weiteren Token erlaubt.

#elif *KonstanterAusdruck*
Zwischen einer **#if**-, **#ifdef**- oder **#ifndef**-Anweisung und einer **#else**- oder **#endif**-Anweisung dürfen beliebig viele **#elif**-Anweisungen stehen. Die Zeilen zwischen der **#elif**- und der nächsten **#else**-, **#elif**- oder **#endif**-Anweisung werden genau dann in die Ausgabe übernommen, wenn die vorhergehende Test-Anweisung und alle folgenden **#elif**-Anweisungen zu Null und *KonstanterAusdruck* zu einem von Null verschiedenen Wert ausgewertet werden. Wenn *KonstanterAusdruck* zu einem von Null verschiedenen Wert ausgewertet wird, werden alle folgenden **#elif**- und **#else**-Anweisungen bis zum dazugehörigen **#endif** ignoriert.

#else Alle zwischen dieser Anweisung und dem dazugehörigen **#endif** stehenden Zeilen werden nur dann in die Ausgabe übernommen, wenn alle vorhergehenden Test-Anweisungen zu Null ausgewertet wurden. Auf derselben Zeile sind keine weiteren Token erlaubt.

#error Ausgeben eines fatalen Fehlers.

#warning
Ausgeben einer Warnung.

ctags

ctags [*Optionen*] *Dateien*

Erzeugen einer Liste von Funktions- und Makronamen, die in den in *Dateien* angegebenen C-, Pascal-, FORTRAN-, **yacc**- oder **lex**-Quelldateien vorkommen. Die Ausgabedatei hat defaultmäßig den Namen **tags** und folgendes Format:

Name *Datei* *Kontext*

wobei *Name* der Funktions- oder Makroname, *Datei* die Quelldatei, in der *Name* definiert wurde, und *Kontext* ein Suchmuster ist, das die *Name* definierende Zeile angibt. Nachdem diese Liste erzeugt wurde, kann **vi** mit einer beliebigen Datei aufgerufen werden. Durch Eingabe von:

```
:set tags=Tagsdatei
:tag name
```

lädt **vi** die Quelldatei, in der *Name* laut *Tagsdatei* definiert ist. Die Quelldatei kann auch auf der Kommandozeile mit `-f` angegeben werden.

etags erzeugt eine entsprechende Datei, die mit dem Emacs verwendet werden kann.

Optionen

ctags

-a, --**append**
: Anhängen der Ausgabe an eine bereits existierende Tag-Liste.

-d, --**defines**
: Auch Tag-Einträge für C-Präprozessor-Definitionen ausgeben.

-o *Datei*, --**output**=*Datei*
: Schreiben der Ausgabe nach *Datei*.

-t, --**typedefs**
: Tag-Einträge für **typedefs** schreiben.

-u, --**update**
: Aktualisieren der Tags-Datei, so daß die veränderten Orte von Funktionen wiedergegeben werden. Alte Tags werden gelöscht, neue angehängt. Diese Option kann z.B. verwendet werden, wenn Funktionen in eine andere Quelldatei verschoben worden sind.

-v, --**vgrind**
: Ausgeben eines Index-Listings nach Standardausgabe, das jede Funktion mit Quelldatei und Seitennummer enthält (eine Seite entspricht 64 Zeilen).

-w, --**no-warn**
: Unterdrücken von Warnungen.

-x, --**cxref**
: Ausgeben eines Listings, in dem für jede Funktion Zeilennummer, Quelldatei und Kontext angegeben werden.

-B, --**backward-search**
: Durchsucht die Dateien vom Ende zum Anfang nach Tags.

-C, --**c++**
: Erwartet, daß *.c*- und *.h*-Dateien C++-Code (anstelle von C-Code) enthalten.

-S, --**ignore-indentation**
: Normalerweise verwendet **ctags** Einrückungen, um die Tag-Datei zu parsen. Mit dieser Option verläßt sich **ctags** weniger darauf.

-T, --**typedefs-and-c++**
: Ausgeben von Tag-Einträgen für typedefs, structs, enums, unions und C++-Memberfunktionen.

---

etags [*Optionen*] *Dateien*

etags

Erzeugen einer Liste von Funktions- und Makronamen, die in den in *Dateien* angegebenen C-, Pascal-, FORTRAN-, **yacc**- oder **lex**-Quelldateien vorkommen. Die Ausgabedatei hat defaultmäßig den Namen **tags** und folgendes Format:

Name *Datei* *Kontext*

wobei *Name* der Funktions- oder Makroname, *Datei* die Quelldatei, in der *Name* definiert wurde, und *Kontext* ein Suchmuster ist, das die *Name* definierende Zeile angibt. Nachdem diese Liste erzeugt wurde, kann Emacs mit einer beliebigen Datei aufgerufen werden.

→

etags
←

Nach Eingabe von:

```
ESC-x visit-tags-table
```

fragt Emacs nach dem Namen einer Tag-Datei; der Default ist TAGS. Jetzt kann mit

```
ESC-x find-tag
```

in die Datei gewechselt werden, die laut *Tagsdatei* einen bestimmten *Namen* enthält. Emacs fragt nach dem zu suchenden Tag. **ctags** erzeugt eine entsprechende Datei, die mit **vi** verwendet werden kann.

Optionen

-a, --**append**
: Anhängen der Ausgabe an eine bereits existierende Tag-Liste.

-i *Datei*, --**include=***Datei*
: Zusätzlich zur normalen Eingabedatei wird noch *Datei* eingelesen.

-o *Datei*, --**output=***Datei*
: Schreiben der Ausgabe nach *Datei*.

-C, --**c++**
: Erwartet, daß **.c**- und **.h**-Dateien C++-Code (anstelle von C-Code) enthalten.

-D, --**no-defines**
: Unterdrücken der Ausgabe von Tag-Einträgen für C-Präprozessor-Definitionen.

-S, --**ignore-indentation**
: Normalerweise verwendet **etags** Einrückungen, um die Tag-Datei zu parsen. Mit dieser Option verläßt sich **etags** weniger darauf.

---

flex

flex [*Optionen*] [*Datei*]

flex (Fast Lexical Analyzer Generator) ist eine schnellere Variante von **lex**. Aus regulären Ausdrücken und C-Anweisungen, die in einer oder mehreren Eingabedateien stehen, erzeugt **flex** ein Programm zur lexikalischen Analyse mit dem Namen *lex.yy.c*. Siehe auch **bison**, **yacc** und *lex & yacc* von John Levine, Tony Mason und Doug Brown.

Optionen

-b Erzeugen von Backup-Informationen in *lex.backup*.

-d Debugging-Modus.

-f Schnellere Übersetzung (nur bei kleinen Programmen).

-h Zusammenfassung der Optionen.

-i Scannen ohne Berücksichtigung von Groß- und Kleinschreibung.

-l Größtmögliche Kompatibilität mit **lex**.

-o *Datei*
: Schreiben der Ausgabe nach *Datei* anstatt nach *lex.yy.c*.

-p Ausgeben eines Performanz-Berichts.

- **-s** Abbruch, wenn der Scanner auf Eingabesymbole trifft, die zu keiner seiner Regeln passen.
- **-t** Schreiben der Ausgabe nach Standardausgabe.
- **-v** Ausgeben einer Statistik.
- **-w** Unterdrücken von Warnungen.
- **-B** Erzeugen eines nicht-interaktiven Scanners, der im Batch-Betrieb arbeitet.
- **-F** Verwenden einer schnellen Tabellen-Repräsentation für den Scanner.
- **-I** Erzeugen eines interaktiven Scanners (Default).
- **-L** Unterdrücken von **#line**-Anweisungen in *lex.yy.c*.
- **-P** *Präfix*
  Verwenden von *Präfix* anstelle des Defaults **yy** für alle global sichtbaren Variablen und Funktionen.
- **-V** Ausgabe der Versionsnummer.
- **-7** Erzeugen eines 7-Bit-Scanners.
- **-8** Erzeugen eines 8-Bit-Scanners (Default).
- **-+** Erzeugen einer C++-Scanner-Klasse.
- **-C** Scanner-Tabellen komprimieren, aber keine Äquivalenzklassen erzeugen.
- **-Ca**
  Tabellen ausrichten. Dies bringt einen schnelleren Zugriff auf Kosten eines höheren Speicherverbrauchs.
- **-Ce**
  Erzeugen von Äquivalenz-Klassen. Dies bringt kleinere Tabellen bei kleinen Performanzeinbußen (Default).
- **-Cf**
  Erzeugen von vollen, nicht komprimierten Scanner-Tabellen.
- **-CF**
  Erzeugen von schnellen Scanner-Tabellen. Ähnlich wie **-F**.
- **-Cm**
  Erzeugen von Meta-Äquivalenz-Klassen (Default).
- **-Cr**
  read()-Systemaufrufe anstatt der Funktionen der Standard-I/O-Bibliothek verwenden.

---

g++ [*Optionen*] *Dateien* — g++

Aufrufen von **gcc** mit den zur Übersetzung von C++-Quelldateien notwendigen Optionen. **g++** erkennt die gleichen Dateiextensionen wie **gcc** sowie zusätzlich C++-Quelldateien (**.C**, **.cc** und **.cxx** Dateien) und bereits vom Präprozessor verarbeitete C++-Dateien (*.ii* Dateien). Siehe auch **gcc**.

---

gcc [*Optionen*] *Dateien* — gcc

Kompiliert eine oder mehrere C-Quelldateien (*Datei*.c), Assembler-Quelldateien (*Datei*.s) oder bereits vom Präprozessor verarbeitete

→

Programmieren

Quelldateien (*Datei*.i). Wenn das Suffix der Datei nicht bekannt ist, wird angenommen, daß es sich um eine Objektdatei oder eine Bibliothek handelt. **gcc** ruft automatisch den Linker **ld** auf, es sei denn, eine der Optionen **-c**, **-S** oder **-E** wurde angegeben. Wenn eine Objektdatei erzeugt wird, hat diese das Suffix **.o** und den gleichen Basisnamen wie die Quelldatei. Defaultmäßig wird die Ausgabe sonst nach **a.out** geschrieben. **gcc** kennt viele systemspezifische Optionen, die hier nicht besprochen werden.

Anmerkung: **gcc** ist die GNU-Variante von **cc**. Auf den meisten Linux-Systemen wird bei Eingabe von **cc** ebenfalls der **gcc** aufgerufen. Der Befehl **g++** ruft den **gcc** mit den zur Übersetzung von C++-Quelltexten notwendigen Optionen auf.

Optionen

-a Erzeugen von Profiler-Informationen für Anweisungsblöcke.

-ansi Erzwingen voller Konformanz mit dem ANSI-Standard.

-b *machine*
Erzeugen von Maschinencode für einen Rechner des Typs *machine*.

-c Erzeugen einer linkbaren Objektdatei für jede Quelldatei; der Linker wird nicht aufgerufen.

-dD Ausgeben von mit **#define** definierten Symbolen.

-dM Unterdrücken der normalen Ausgabe. Statt dessen werden alle **#defines** ausgegeben, die nach dem Ende des Präprozessoraufrufs definiert werden.

-dN Nur die Makronamen von **#defines** werden ausgegeben, nicht die Argumente oder Werte.

-fno-ident
#ident-Anweisungen werden nicht beachtet.

-fsigned-char
Der Datentyp **char** wird als vorzeichenlos interpretiert.

-fsyntax-only
Es wird nur die Syntax des Quelltextes überprüft, aber keine Objektdatei erzeugt.

-funsigned-char
Der Datentyp **char** wird als vorzeichenbehaftet interpretiert.

-g Einfügen von Debugging-Informationen zur Verwendung mit **gdb** in die Objektdatei.

-glevel *Stufe*
Einfügen von Debugging-Informationen der Stufe *Stufe*. Gültige Werte sind 1, 2 und 3, wobei 1 den niedrigsten Informationsgehalt bezeichnet. Der Default ist 2.

-gxcoff
Erzeugen von Debugging-Informationen im XCOFF-Format, die mit dem **dbx**-Debugger verwendet werden können.

-idirafter *Verz*
Das Verzeichnis *Verz* wird durchsucht, wenn eine Header-Datei nicht im normalen Include-Pfad gefunden werden konnte.

-include *Datei*
: Vor der Verarbeitung der Eingabedateien wird zunächst *Datei* verarbeitet.

-imacros *Datei*
: Vor der Verarbeitung der Eingabedateien werden zunächst die Makros in *Datei* verarbeitet.

-iprefix *Präfix*
: Wenn Verzeichnisse dem Suchpfad mit **-iwithprefix** hinzugefügt werden, wird jedem dieser Verzeichnisse *Präfix* vorangestellt.

-iwithprefix *Verz*
: Das Verzeichnis *Verz* wird der Liste der Verzeichnisse hinzugefügt, die durchsucht werden, wenn eine Header-Datei nicht im normalen Include-Pfad gefunden werden konnte. Wenn vorher **-iprefix** verwendet wurde, wird dieser Wert dem Verzeichnisnamen vorangestellt.

-l*lib* Linken mit Bibliothek **lib**.

-lobjc Unterstützung zum Linken von Objective C-Programmen.

-m486 Optimieren des Codes für Intel 80486-Rechner (und höher).

-mno-486
: Optimieren des Codes für Intel 80386-Rechner (anstelle von 80486 und höher).

-mno-fp-ret-in-387
: Keine FPU-Register für Rückgabewerte von Funktionen verwenden.

-nostartfiles
: Der Linker ignoriert die standardmäßigen System-Startdateien.

-nostdinc
: Header-Dateien werden nur in den angegebenen, aber nicht in den Standard-Verzeichnissen gesucht.

-nostdinc++
: Header-Dateien werden nicht in Verzeichnissen durchsucht, in denen C++-spezifische Header-Dateien vermutet werden.

-nostdlib
: Nicht mit den Standard-Bibliotheken linken.

-o *Datei*
: Schreiben der Ausgabe nach *Datei*. Der Default ist *a.out*.

-pedantic
: Besonders viele Warnungen ausgeben.

-pedantic-errors
: Ausgeben eines Fehlers (und damit kein Erzeugen einer Objektdatei) in allen Fällen, in denen **-pedantic** eine Warnung ausgegeben hätte.

-pg Einfügen von Profiling-Informationen zur Verwendung mit **gprof**.

-pipe Die temporären Informationen zwischen den Compilerläufen werden über Pipes anstelle von temporären Dateien übertragen.

→

-print-file-name=*Bibliothek*
Unterdrücken der normalen Ausgabe. Es wird lediglich der absolute Pfadname der Bibliothek ausgegeben, die zur ausführbaren Datei hinzugelinkt werden würde.

-static Statische anstelle von dynamischen Bibliotheken erzeugen.

-traditional
Weitestgehendes Imitieren des Verhaltens traditioneller C Compiler.

-traditional-cpp
Weitestgehendes Imitieren des Verhaltens traditioneller C-Präprozessoren.

-trigraphs
Unterstützung für Trigraphen einschalten.

-u *Symbol*
Erzwingen, daß der Linker in den Bibliotheken nach dem Symbol *Symbol* sucht und diese Bibliotheken dazulinkt, falls das Symbol gefunden wurde.

-v „Verbose-Modus". Angezeigt werden die ausgeführten Befehle, die Versionsnummer des **gcc** und die Versionsnummer des Präprozessors.

-w Unterdrücken von Warnungen.

-x *Sprache*
Erwartet, daß die Quelldatei in der Sprache *Sprache* geschrieben ist. Gültige Werte für *Sprache* sind: **c**, **objective-c**, **c-header**, **c++**, **cpp-output**, **assembler**, und **assembler-with-cpp**. Wenn **none** angegeben wird, wird die Sprache anhand der Dateiextension geraten.

-A*Bedingung*(*Antwort*)
Wenn der Präprozessor auf einen konditionalen Ausdruck wie **#if** *Bedingung* stößt, wird *Antwort* angenommen. Die Standard-Antworten können mit **-A-** abgeschaltet werden.

-C Kommentare bleiben während des Präprozessierens erhalten. Nur sinnvoll mit **-E**.

-D*Name*[*=Def*]
Definiert das Symbol *Name* mit dem Wert *Def*, genau wie mit einer **#define**-Anweisung. Wenn *=Def* nicht angegeben wird, wird 1 als Wert angenommen. **-D** hat eine niedrigere Priorität als **-U**.

-E Quelldateien werden präprozessiert, aber nicht kompiliert. Das Ergebnis wird auf Standardausgabe ausgegeben.

-I*Verz* Einfügen von *Verz* in die Liste der Verzeichnisse, in denen nach Header-Dateien gesucht wird. Wenn *Verz* - ist, werden die mit **-I** vor dem *-I-* angegebenen Verzeichnisse nur durchsucht, wenn **#include "Datei"** anstelle von **#include <Datei>** angegeben wurde.

-L*Verz* Durchsuchen von *Verz* nach Bibliotheken zusätzlich zu den Standard-Verzeichnissen.

-M, -MG
Unterdrücken der normalen Ausgabe. Es werden **make**-Regeln ausgegeben, welche die Abhängigkeiten der Quelldatei beschreiben. Impliziert **-E**.

-MM [-MG]
Ähnlich wie **-M**. Es werden nur für die Dateien Abhängigkeiten angegeben, die mit **#include** "*Datei*" eingefügt wurden. Impliziert **-E**.

-H Ausgeben der eingefügten Dateien (eine pro Zeile) nach Standardausgabe.

-O[*Stufe*]
Optimieren der Objektdateien. *Stufe* kann einer der Werte 1, 2, 3 oder 0 sein. Der Default ist 1. 0 schaltet die Optimierung ab, 3 ist die höchste Optimierungsstufe.

-P Präprozessiert die Eingabe ohne Erzeugung von Zeileninformationen, die im nächsten Durchlauf verwendet werden könnten. Nur sinnvoll zusammen mit **-E**.

-S Erzeugen von Assembler-Dateien aus den Quelldateien, aber kein Aufrufen des Assemblers.

-U*Name*
Entfernen von bisher bekannten Definitionen des Symbols *Name*, wobei *Name* ein reserviertes Symbol des Präprozessors oder ein mit **-D** definiertes Symbol sein kann. Auf Intel-Rechnern definierte Symbole sind **unix** und **i386**.

-V *Version*
Wenn vorhanden, wird der **gcc** mit der angegebenen *Version* gestartet.

-W Ausgeben von mehr Warnungen als normal.

-Wl,*Option*
Aufrufen des Linkers mit *Option*. *Option* kann eine mit Kommata getrennte Liste von Optionen sein.

-Wa,*option*
Aufrufen des Assemblers mit *option*. *option* kann eine mit Kommata getrennte Liste von Optionen sein.

-Waggregate-return
Ausgeben einer Warnung, wenn Funktionen definiert oder aufgerufen werden, die structs oder unions zurückgeben.

-Wall Schaltet **-W**, **-Wchar-subscripts**, **-Wcomment**, **-Wformat**, **-Wimplicit**, **-Wparentheses**, **-Wreturn-type**, **-Wswitch**, **-Wtemplate-debugging**, **-Wtrigraphs**, **-Wuninitialized** und **-Wunused** ein.

-Wcast-align
Ausgeben einer Warnung, wenn Pointer auf einen Typ gecastet werden, so daß die notwendige Ausrichtung größer wird als ursprünglich.

-Wcast-qual
Ausgeben einer Warnung, wenn Pointer auf Typen gecastet werden, die nicht die gleichen Typattribute wie das Original haben.

→

-Wchar-subscripts
Ausgeben einer Warnung, wenn Arrays mit Indizes vom Typ **char** definiert oder verwendet werden.

-Wcomment
Ausgeben einer Warnung, wenn der Beginn eines verschachtelten Kommentars gefunden wird.

-Wconversion
Ausgeben einer Warnung bei bestimmten Typkonversionen.

-Werror
Abbruch beim ersten Auftreten eines Fehlers.

-Wformat
Ausgeben einer Warnung bei inkorrekt formatierten **printf**- und **scanf**-Aufrufen.

-Wimplicit
Ausgeben einer Warnung, wenn Funktionen oder Parameter implizit definiert werden.

-Winline
Ausgeben einer Warnung bei illegalen Inline-Funktionen.

-Wmissing-declarations
Ausgeben einer Warnung, wenn eine globale Funktion definiert wird, ohne vorher deklariert worden zu sein.

-Wmissing-prototypes
Ausgeben einer Warnung, wenn globale Funktionen definiert werden, ohne daß vorher ein Prototyp deklariert wurde.

-Wnested-externs
Ausgeben einer Warnung, wenn eine **extern**-Deklaration innerhalb einer Funktion gefunden wurde.

-Wno-import
Keine Warnung ausgeben, wenn **#import** verwendet wird.

-Wparentheses
Mehr Warnungen bezüglich vergessener Klammern ausgeben.

-Wpointer-arith
Ausgeben einer Warnung bei Auftreten von Code, der versucht, die Größe einer Funktion oder von **void** zu bestimmen.

-Wredundant-decls
Ausgeben einer Warnung, wenn etwas öfter als einmal in einem Gültigkeitsbereich deklariert wird.

-Wreturn-type
Ausgeben einer Warnung, wenn Funktionen ohne oder mit falschen Rückgabewerten definiert werden.

-Wshadow
Ausgeben einer Warnung, wenn eine lokale Variable eine andere überschattet.

-Wstrict-prototypes
Nur Code akzeptieren, in dem alle Argumenttypen in Funktionsdeklarationen und -definitionen spezifiziert werden.

-Wswitch
Ausgeben einer Warnung bei switch-Anweisungen bezüglich Aufzählungstypen, in denen nicht alle Werte des Aufzählungstyps auftreten.

-Wtemplate-debugging
Ausgeben einer Warnung, wenn Debugging-Informationen für C++-Templates nicht zur Verfügung stehen.

-Wtraditional
Ausgeben einer Warnung, wenn Code verwendet wird, der in ANSI-C und K&R-C unterschiedlich interpretiert wird.

-Wtrigraphs
Ausgeben einer Warnung bei Trigraphen.

-Wuninitialized
Ausgeben einer Warnung, wenn uninitialisierte automatische Variablen verwendet werden.

-Wunused
Ausgeben einer Warnung, wenn eine Variable oder Funktion nicht benutzt wird.

Pragma-Anweisungen

#pragma interface [*Header-Datei*]
Wird in Header-Dateien benutzt und erzwingt, daß die Definitionsinformation in den Objektdateien als Referenz und nicht als lokale Kopie aufgenommen wird. C++-spezifisch.

#pragma implementation [*Header-Datei*]
Wird in normalen Quelldateien verwendet und erzwingt die Ausgabe vollständiger Information aus der Header-Datei *Header-Datei* (oder, wenn *Header-Datei* nicht angegeben wird, aus der Header-Datei, die denselben Basisnamen wie die Datei hat, welche die Pragma-Anweisung enthält). Diese Information ist global sichtbar. Normalerweise enthält die angegebene Header-Datei eine **#pragma interface**-Anweisung.

gdb [*Optionen*] [*Programm* [*Core*| *PID*]]

GDB, der GNU DeBugger, ermöglicht es, in einzelnen Schritten durch ein C-, C++- oder Modula-2-Programm zu laufen, um so herauszufinden, an welcher Stelle ein Fehler auftritt. Das Programm, das untersucht werden soll, wird üblicherweise als Kommandozeilenargument angegeben. Außerdem kann eine Core-Datei oder eine Prozeßnummer (wenn ein bereits laufendes Programm durch einen Debugger geschickt werden soll) angegeben werden.

Optionen

-s *Datei*, **-symbols=***Datei*
Laden der Symboltabelle aus der *Datei.* Wird auch **-e** angegeben, ist diese Datei auch das Programm, das durch den Debugger soll.

→

gdb
←

-e *Datei*, **-exec=***Datei*
: Debuggen der Datei *Datei*. Wenn zusätzlich **-s** angegeben wird, wird auch die Symboltabelle eingelesen.

-c *Datei*, **-core=***Datei*
: Suchen von Core-Dump-Informationen in der Datei *Datei*.

-x *Datei*, **-command=***Datei*
: Einlesen und Ausführen von **gdb**-Befehlen aus der Datei *Datei*.

-d *Verzeichnis*, **-directory=** *Verzeichnis*
: Quelldateien auch im Verzeichnis *Verzeichnis* suchen.

-n, -nx
: Die Datei *.gdbinit* nicht berücksichtigen.

-q, -quiet
: Start- und Copyright-Meldungen unterdrücken.

-batch Ausführen aller in *.gdbinit* und mit **-x** angegebenen Befehle ohne Ausgeben von Start-Meldungen. Nach der Ausführung beendet sich **gdb** wieder.

-cd= *Verzeichnis*
: *Verzeichnis* als aktuelles Arbeitsverzeichnis von **gdb** verwenden.

-f, -fullname
: Anzeigen des vollen Pfadnamens und der Zeilennummer in jedem Stack Frame.

-b *bps* Einstellen der Geschwindigkeit der seriellen Schnittstelle, die vom GDB benutzt wird.

-tty=*device*
: Einstellen der Standardeingabe und -ausgabe auf *device*.

Gebräuchliche Befehle

Die folgenden Befehle des GDB werden besonders häufig verwendet. Sie finden hier nur eine Auswahl, da es viel zu viele Befehle im GDB gibt, um alle hier aufführen zu können.

backtrace
: Ausgeben der aktuellen Stelle im Programm sowie eines Stack Traces, der anzeigt, wie das Programm an die aktuelle Stelle gekommen ist. (**where** macht das gleiche.)

breakpoint
: Setzen eines Haltepunktes.

cd Wechseln des aktuellen Verzeichnisses.

clear Löschen des Haltepunktes, an dem das Programm zuletzt angehalten hat.

commands
: Ausgeben der Befehle, die ausgeführt werden, wenn das Programm auf einen Haltepunkt stößt.

continue
: Fortsetzen der Programmausführung nach dem Erreichen eines Haltepunktes.

delete Löschen eines Haltepunktes oder eines Überwachungspunktes; wird auch in Verbindung mit anderen Befehlen benutzt.

display Anzeigen von Variablen oder Ausdrücken, wenn ein Haltepunkt erreicht wird.

down Abwärts im Stack bewegen. Der erreichte Stack Frame wird der aktuelle.

frame Auswahl eines Frames für den nächsten **continue**-Befehl.

info Ausgeben von Informationen über das Programm. **info breakpoints** zeigt beispielsweise alle Halte- und Überwachungspunkte an.

jump Fortsetzen der Ausführung an einer anderen Stelle in der Quelldatei.

kill Abbrechen des von **gdb** kontrollierten Prozesses.

list Ausgeben eines Teils der Quelldatei, die zu der aktuellen Stelle im Programm gehört.

next Ausführen der nächsten Zeile Quelltext. Wenn es sich um einen Funktionsaufruf handelt, wird diese Funktion vollständig ausgeführt.

print Ausgeben des Wertes einer Variablen oder eines Ausdrucks.

pwd Ausgeben des aktuellen Arbeitsverzeichnisses.

ptype Ausgeben des Inhalts eines Datentyps (z.B. einer Struktur oder einer C++-Klasse).

quit Beenden von **gdb**.

reverse-search
Rückwärts nach einem regulären Ausdruck in der Quelldatei suchen.

run Starten des Programms.

search Vorwärts nach einem regulären Ausdruck in der Quelldatei suchen.

set variable
Zuweisen eines Wertes an eine Variable.

signal Senden eines Signals an den laufenden Prozeß.

step Ausführen der nächsten Zeile des Quelltextes. Wenn es sich um einen Funktionsaufruf handelt, wird in die Funktion hineingesprungen und dort angehalten.

undisplay
Aufheben des **display**-Befehls; Ausdrücke werden nicht mehr angezeigt.

until Beenden der aktuellen Schleife.

up Aufwärts im Stack bewegen. Der erreichte Stack Frame wird der aktuelle.

watch Setzen eines Überwachungspunktes *(watchpoint)*, also eines Haltepunktes auf Daten.

whatis Ausgeben des Typs einer Variablen oder einer Funktion.

gprof

gprof [*Optionen*] [*Objektdatei*]

Anzeigen der Profiling-Daten einer Objektdatei. Die Symboltabelle der Datei wird mit dem Profil des Aufrufgraphen in der Datei *gmon.out* verglichen, die beim Kompilieren mit **gcc -pg** angelegt wurde.

Optionen

-a Statisch deklarierte Funktionen nicht ausgeben. Da die ihnen zugeordnete Information für die Profiling-Daten trotzdem relevant sein könnte, werden diese an die Information der unmittelbar zuvor geladenen Funktion angehängt.

-b Keine Information über jedes Feld anzeigen.

-c Verwenden der text section der Objektdatei, um den statischen Aufrufgraphen des Programms zu bestimmen. Statische Eltern und Kinder werden mit null Aufrufen angezeigt.

-e *Routine*
Keine Einträge für *Routine* und deren Kinder anzeigen.

-f *Routine*
Nur *Routine* ausgeben. Die Zeit, die in anderen Teilen verbracht wurde, wird *Routine* zugeschlagen.

-k *von nach*
Entfernen der Kanten zwischen den Routinen *von* und *nach*.

-s Zusammenfassen der Profiling-Information in der Datei *gmon.sum*.

-v Ausgeben der Versionsnummer und Beenden von **gprof**.

-z Auch nicht verwendete Aufrufe ausgeben.

-E *Routine*
Keine Einträge für *Routine* und deren Kinder anzeigen. Die in diesen Programmteilen verbrachte Zeit wird bei der Berechnung der Gesamtzeit ebenfalls vernachlässigt.

-F *Routine*
Nur *Routine* ausgeben. Die Zeit, die in anderen Teilen verbracht wurde, wird vernachlässigt.

imake

imake *Optionen*

imake (*include make*) ist eine **cpp**-Schnittstelle zum **make**-Werkzeug und beseitigt die üblichen Portabilitätsprobleme von **make**, indem Systemabhängigkeiten in einem zentralen Satz von Konfigurationsdateien aufgeführt werden, die somit von der Beschreibung der zu erzeugenden Programme getrennt sind. Die **make**-Ziele werden in einer Datei *Imakefile* systemunabhängig in Form von **cpp**-Makros beschrieben. **imake** verarbeitet die Konfigurationsdateien und das *Imakefile* mit **cpp** und erzeugt daraus systemabhängige *Makefiles*, die danach von **make** verwendet werden können.

Eine der Konfigurationsdateien ist eine Template-Datei, die Masterdatei von **imake**. Diese Template-Datei, die defaultmäßig *Imake.tmpl* heißt, lädt die anderen Konfigurationsdateien, die systemabhängige Beschreibungen wie Variablenzuweisungen, lokale Einstellungen und **cpp**-Makros enthalten, per **#include**-Anweisung und steuert die Reihenfolge, in der diese Dateien

verarbeitet werden. Jede Datei beeinflußt die Verarbeitung später eingefügter Dateien und nachfolgender Abschnitte in *Imake.tmpl.*

In **imake**-Konfigurationsdateien können Kommentare verwendet werden, aber dem einleitenden # muß ein leerer C-Kommentar vorangestellt werden:

```
/**/#
```

Weitere Informationen finden Sie unter **cpp** und **make** sowie im Nutshell-Handbuch *Software Portability with imake* von Paul DuBois.

Optionen

-D*Definition*
: Setzen verzeichnis-spezifischer Variablen. Diese Option wird direkt an **cpp** weitergegeben.

-e
: Ausführen des erzeugten *Makefile.* Defaultmäßig wird dies dem Benutzer überlassen.

-f *Dateiname*
: Angeben des Namens der verzeichnis-spezifischen Eingabedatei. Der Default ist **Imakefile**.

-I*Verzeichnis*
: Angeben des Verzeichnisses, in dem die Template-Datei und die anderen Konfigurationsdateien zu finden sind. Diese Option wird direkt an **cpp** weitergegeben.

-s *Dateiname*
: Angeben des Namens des zu erzeugenden *Makefile.* Wenn *Dateiname* ein Querstrich ist, wird die Ausgabe auf die Standardausgabe geschrieben. Defaultmäßig heißt die erzeugte Datei *Makefile.*

-T*Template*
: Angeben der von **cpp** verwendeten Template-Datei. Diese Datei steht üblicherweise in einem Verzeichnis, das mit **-I** angegeben wurde. Der Default ist *Imake.tmpl.*

-v
: Ausgeben der **cpp**-Kommandozeile, mit der das **Makefile** erzeugt wird.

Werkzeuge

Im folgenden finden Sie eine Liste von Werkzeugen, die von **imake** verwendet werden:

makedepend [*Optionen*] *Dateien*
: Erzeugen der Abhängigkeiten von Header-Dateien in *Dateien.* **makedepend** liest die übergebenen Dateien ein und sucht nach **#include**-, **#define**-, **#undef**-, **#ifdef**-, **#ifndef**-, **#endif**-, **#if**- und **#else**-Anweisungen, um zu bestimmen, welche **#include**-Anweisungen während der Übersetzung tatsächlich zum Tragen kommen. **makedepend** bestimmt daraus rekursiv die Abhängigkeiten und schreibt diese in das **Makefile**. Diese Angabe wird von **make** verwendet, um zu bestimmen, welche Dateien neu übersetzt werden müssen, wenn eine Header-Datei verändert wurde. **makedepend** kennt die folgenden Optionen:

→

-- *Optionen* --
: Ignorieren aller unbekannten Optionen nach dem doppelten Querstrich. Ein zweiter doppelter Querstrich beendet diesen Zustand. Bekannte Optionen zwischen den beiden doppelten Querstrichen werden normal verarbeitet.

-a
: Anhängen der Abhängigkeiten an die existierenden, anstatt sie zu ersetzen.

-f*Dateiname*
: Ausgeben der Abhängigkeiten in die Datei mit dem Namen *Dateiname*, anstatt sie an **Makefile** anzuhängen.

-m
: Ausgeben einer Warnung, wenn eine Header-Datei mehrfach eingelesen wird.

-s*String*
: Verwenden von *String* als Abgrenzung im *Makefile*. Der Default ist **# DO NOT DELETE THIS LINE — make depend depends on it**.

-v
: „verbose-Modus". Ausgeben aller Dateien, die von der untersuchten Quelldatei eingelesen werden.

-D*Name*=*Wert*

-D*Name*
: Definieren des Symbols *Name* mit dem angegebenen Wert (erste Variante) oder mit 1 (zweite Variante).

-I*Verz*
: Hinzufügen des Verzeichnisses *Verz* zur Liste der zu durchsuchenden Verzeichnisse.

-Y*Verz*
: Nur in *Verz* nach Header-Dateien suchen. Die Standard-Verzeichnisse werden ignoriert.

mkdirhier *Verz*...
: Erzeugen des Verzeichnisses *Verz* und aller fehlenden Eltern-Verzeichnisse während der Installation von Dateien.

xmkmf [*Option*] [*WurzelVerz*] [*AktVerz*]
: Erstmaliges Erzeugen eines *Makefile* aus einem *Imakefile*. *WurzelVerz* gibt den Ort des Wurzelverzeichnisses des Projekts an. *AktVerz* wird normalerweise weggelassen und bezeichnet einen relativen Dateinamen vom Beginn des Quelltextbaumes zum aktuellen Verzeichnis. Die Option **-a** entspricht der folgenden Befehlsfolge:

```
% xmkmf
% make Makefiles
% make includes
% make depend
```

Konfigurationsdateien

Im folgenden finden Sie eine Liste der von **imake** verwendeten Konfigurationsdateien.

Imake.tmpl
: Haupt-Template von **imake**. *Imake.tmpl* lädt alle anderen Konfigurationsdateien sowie die Datei *Imakefile* im aktuellen Verzeichnis nach.

Imake.rules

Enthält **cpp**-Makros, die für die jeweilige Plattform geschrieben wurden. Die Makros werden von **imake** mit Hilfe des **cpp** verarbeitet. Fortsetzungszeilen werden durch die Zeichenkette @@\ (zwei Klammeraffen und ein Backslash) angezeigt.

site.def Enthält Informationen, die für die lokale Installation spezifisch sind (und nicht für den jeweiligen Hersteller des Systems). Dazu gehören beispielsweise Installationsverzeichnisse, welche Programme erzeugt werden sollen und welche besonderen Programme oder Programmversionen während des Übersetzungsprozesses verwendet werden sollen. Die Datei *site.def* ist auf jedem Rechner anders.

Project.tmpl

Enthält X spezifische Variablen.

Library.tmpl

Enthält Regeln zur Verwendung und Erzeugung von Bibliotheken.

Server.tmpl

Enthält server-spezifische Regeln.

cf Bei den *.cf*-Dateien handelt es sich um die hersteller-abhängigen sogenannten *VendorFiles*, die in *Imake.vb* zu finden sind. Eine *.cf*-Datei enthält plattformspezifische Definitionen wie Versionsnummern des Betriebssystems oder des Compilers sowie Workarounds für auf dieser Plattform nicht vorhandene Befehle. Die Definitionen in den *.cf*-Dateien haben Vorrang vor den Defaults in *Imake.params*.

Die Datei Imakefile

Die Datei *Imakefile* ist eine Datei, die in jedem Verzeichnis zu finden ist und angibt, welche Dateien zu erzeugen und zu installieren sowie welche Regeln dazu anzuwenden sind. **imake** liest die Datei *Imakefile* ein und expandiert die dort zu findenden Regeln in Einträge in der Datei *Makefile*. Ein *Imakefile* kann außerdem **make**-Variablendefinitionen und Abhängigkeiten enthalten. Die Abhängigkeiten werden in Form von **cpp-Makros** beschrieben, die in *Imake.rules* definiert sind. Wenn Sie das *Imakefile* verändern, müssen Sie das zugehörige *Makefile* und die Abhängigkeiten neu erzeugen.

ld [*Optionen*] *Objektdateien*

ld faßt mehrere *Objektdateien* in der angegebenen Reihenfolge in einer einzelnen ausführbaren Datei zusammen, die defaultmäßig *a.out* heißt. **ld** ist der System-Linker und wird meistens automatisch vom Compiler aufgerufen.

Optionen

-c *Datei*

Einlesen und Ausführen von Linker-Befehlen aus der Datei *Datei*.

→

ld
←

-d, -dc, -dp
: Diese drei Optionen sind synonym. Sie weisen häufig verwendeten Symbolen auch dann Platz zu, wenn eine relozierbare Datei erzeugt wird.

-defsym *Symbol* = *Ausdruck*
: Erzeugen des globalen Symbols *Symbol* mit dem Wert *Ausdruck*.

-e *Symbol*
: *Symbol* als Adresse des Einsprungpunktes der erzeugten Datei verwenden.

-i
: Erzeugen einer linkbaren Ausgabedatei; die Magic Number wird, wenn möglich, auf OMAGIC gesetzt.

-l*Arch*
: Aufnehmen der Archivdatei oder dynamischen Bibliothek *Arch* in die Liste der zu linkenden Dateien.

-m *Linker*
: Emulation von *Linker*.

-n
: Die Text Section der Ausgabedatei auf Nur-Lesen setzen. Die Magic Number wird, wenn möglich, auf NMAGIC gesetzt.

-noinhibit-exec
: Auch wenn beim Linken Fehler aufgetreten sind, wird eine Ausgabedatei erzeugt.

-o *Datei*
: Die Ausgabe wird nach *Datei* anstelle von *a.out* geschrieben.

-oformat *Format*
: Angabe des Formates der Ausgabe.

-r
: Erzeugen einer linkbaren Ausgabedatei; die Magic Number wird, wenn möglich, auf QMAGIC gesetzt.

-s
: Keine Symbolinformationen in der Ausgabe aufnehmen.

-sort-common
: Globale allgemeine Symbole werden nicht nach Größe sortiert.

-t
: Ausgeben des Namens jeder Eingabedatei, während sie verarbeitet wird.

-u *Symbol*
: Deklarieren des Symbols *Symbol* als undefiniert.

-v, **--version**
: Anzeigen der Versionsnummer.

--verbose
: Ausgeben von Informationen über **ld** sowie der Namen der Eingabedateien, während versucht wird, sie zu öffnen.

-warn-common
: Ausgeben einer Warnung, wenn allgemein verwendete Symbole mit anderen Konstrukten kombiniert werden.

-warn-once
: Nur eine Warnung für jedes undefinierte Symbol ausgeben.

-x
: Wird in Verbindung mit -s und -S verwendet und löscht alle lokalen Symbole, die mit L beginnen.

-L *Verz*
: Durchsuchen des Verzeichnisses *Verz* nach Bibliotheken vor den Standardverzeichnissen.

-M Ausgeben einer Linker-Symboltabelle auf Standardausgabe.

-Map *Datei*
: Ausgeben einer Linker-Symboltabelle in die Datei *Datei*.

-N Sowohl Daten- als auch Textsegmente dürfen gelesen und beschrieben werden. Wenn UNIX Magic Numbers unterstützt werden, wird die Ausgabe entsprechend markiert. Das Datensegment wird nicht an einer Seitengrenze ausgerichtet.

-R *Datei*
: Einlesen der Symbolnamen und Adressen aus der Datei *Datei*. *Datei* wird aber nicht reloziert und auch nicht in die Ausgabedatei mit aufgenommen.

-S Keine Symbolinformationen für den Debugger in die Ausgabedatei schreiben.

-Tbss *Adresse*
: Anfang des Bss-Segments der Ausgabedatei auf *Adresse* festlegen.

-Tdata *Adresse*
: Anfang des Data-Segments der Ausgabedatei auf *Adresse* festlegen.

-Ttext *Adresse*
: Anfang des Text-Segments der Ausgabedatei auf *Adresse* festlegen.

-Ur Identisch mit **-r**, außer wenn C++-Programme gelinkt werden. In diesem Fall werden auch Referenzen auf Konstruktoren aufgelöst.

-X Wird in Verbindung mit -s und -S verwendet und löscht alle lokalen Symbole, die mit **L** beginnen.

-V Ausgeben der Versionsnummer und der Emulationen, die bei der **-m**-Option verwendet werden können.

ldd [*Optionen*] *Programme*

ldd

Ausgeben einer Liste dynamischer Bibliotheken, welche die angegebenen Programme anziehen.

Optionen

-v Ausgeben der Versionsnummer von **ldd**.

-V Ausgeben der Versionsnummer des Linkers.

m4 [*Optionen*] [*Makros*] [*Dateien*]

m4

Makro-Prozessor für C- und andere Dateien.

Optionen

-e Interaktiv arbeiten und Interrupts ignorieren.

→

m4 ←

-d*Stufe*, **--debug=***Stufe*
: Einstellen der Debugging-Stufe.

-l*n*, **--arglength=***n*
: Angeben der Länge der Debugging-Ausgabe.

-o *Datei*, **--error-output=***Datei*
: Schreiben der Ausgabe nach *Datei*. Entgegen dem Optionsnamen werden Fehlermeldungen nach Standardfehler geschrieben.

-p, --prefix-built-ins
: Den Namen aller eingebauten Makros **m4_** voranstellen.

-s, --synclines
: **#line**-Anweisungen für den C-Präprozessor einfügen.

-B*n*
: Einstellen der Größe des Rückhalte- und Argumentbuffers auf *n* (der Default ist 4096).

-D*Name*[=*Wert*], **--define=***Name*[=*Wert*]
: Definieren des Symbols *Name* mit dem Wert *Wert* oder mit einem Leerwert, falls *Wert* nicht angegeben wird.

-E, --fatal-warnings
: Alle Warnungen werden als fatal angesehen und die Ausführung nach der ersten abgebrochen.

-F*Datei*, **--freeze-state** *Datei*
: Abspeichern eines eingefrorenen Zustands in der Datei *Datei*, aus der er später wieder geladen werden kann.

-G, --traditional
: Verhalten wie ein traditioneller **m4**; Ignorieren der GNU-Erweiterungen.

-H*n*, **--hashsize=***n*
: Einstellen der Größe der Hashtabelle für die Symbole auf *n* (der Default ist 509).

-I*Verzeichnis*, **--include=***Verzeichnis*
: Include-Dateien auch in *Verzeichnis* suchen.

-Q, --quiet, --silent
: Keine Warnungen ausgeben.

-R*Datei*, **--reload-state** *Datei*
: Einlesen des eingefrorenen Zustands in der Datei *Datei* vor der Programmausführung.

-U*Name*, **--undefine=***Name*
: Aufheben der Symboldefinition von *Name*.

---

make

make [*Optionen*] [*Ziele*] [*Makrodefinitionen*]

Erzeugen eines oder mehrerer sogenannter Ziele anhand der Abhängigkeitsinformationen in der Beschreibungsdatei im aktuellen Verzeichnis. Defaultmäßig heißt diese Datei *makefile* oder *Makefile*. Optionen, Ziele und Makrodefinitionen dürfen in beliebiger Reihenfolge auftreten. Makrodefinitionen werden in folgender Form angegeben:

```
Name=String
```

Weitere Informationen über **make** finden Sie im Nutshell-Handbuch *Managing Projects with make* von Andrew Oram and Steve Talbott.

Optionen

-b
: Ist obsolet und wird ignoriert.

-d, -dd, --**debug**
: Ausgeben detaillierter Informationen zum Debuggen von Makefiles.

-e, --**environment-overrides**
: Umgebungsvariablen haben Vorrang vor Makrodefinitionen in Makefiles.

-f *Dateiname*, --**file** *Dateiname*, --**makefile** *Dateiname*
: *Dateiname* wird als Beschreibungsdatei verwendet; wenn **-** angegeben wird, werden die Informationen von der Standardeingabe gelesen.

-h, --**help**
: Ausgeben der Optionen von **make**.

-i, --**ignore-errors**
: Ignorieren von Fehlercodes aufgerufener Befehle (wie .IGNORE).

-j [*Jobs*], --**jobs** [*Jobs*]
: Wenn möglich, *Jobs* verschiedene Jobs gleichzeitig ausführen. Wenn keine Zahl angegeben wird, wird versucht, so viele Jobs wie möglich auszuführen.

-k, --**keep-going**
: Wenn das aktuelle Ziel nicht erzeugt werden kann, trotzdem versuchen, andere Ziele zu erzeugen.

-l [*Last*], --**load-average** [*Last*], --**max-load** [*Last*]
: Wird im Zusammenhang mit -j verwendet. **make** versucht, die Systemlast unter *Last*, einer Fließkommazahl, zu halten.

-m
: Obsolet, wird ignoriert.

-n, --**just-print**, --**dry-run**, --**recon**
: Ausgeben der Befehle, die ausgeführt werden würden, ohne sie tatsächlich auszuführen (wird zum Testen von Makefiles verwendet).

-o *Datei*, --**old-file** *Datei*, --**assume-old** *Datei*
: *Datei* und andere Ziele, die davon abhängig sind, nicht neu erzeugen.

-p, --**print-data-base**
: Ausgeben der Regeln und Variablen neben der normalen Ausführung.

-q, --**question**
: Ausgeben von 0, wenn nichts neu erzeugt werden muß, ansonsten wird ein von Null verschiedener Wert ausgegeben.

-r, --**no-built-in-rules**
: Keine eingebauten Regeln verwenden.

→

-s, --silent, --quiet
Kommandozeilen nicht anzeigen (wie .SILENT).

-t, --touch
Neu zu erzeugende Dateien auf die aktuelle Uhrzeit setzen, so daß sie als aktuell gelten und nicht neu erzeugt werden.

-v, --version
Anzeigen der Versionsnummer von **make**.

-w, --print-directory
Ausgeben des aktuellen Arbeitsverzeichnisses vor und nach der Ausführung.

--warn-undefined-variables
Ausgeben einer Warnung, wenn ein Makro verwendet wird, ohne definiert zu sein.

-C *Verzeichnis*, **--directory** *Verzeichnis*
Wechseln nach *Verzeichnis* vor dem Beginn der normalen Operationen. Folgende -C-Anweisungen veranlassen einen Verzeichniswechsel in ein relativ zum aktuellen Arbeitsverzeichnis bestimmtes Verzeichnis.

-I *Verzeichnis*, **--include-dir** *Verzeichnis*
Include-Dateien auch in *Verzeichnis* suchen.

-S, --no-keep-going, --stop
Aufheben von **-k**. Dies ist in rekursiven **make**-Aufrufen nützlich.

-W *Datei*, **--what-if** *Datei*, **--new-file** *Datei* **--assume-new** *Datei*
Annehmen, daß *Datei* gerade verändert worden ist.

Format der Beschreibungsdateien

Anweisungen in den Beschreibungsdateien werden Zeile für Zeile interpretiert. Wenn sich eine Anweisung über mehrere Zeilen erstrecken soll, muß am Ende aller Zeilen mit Ausnahme der letzten ein Backslash (\) stehen, damit die nachfolgenden Zeilen als Folgezeilen interpretiert werden. Die Beschreibungsdateien dürfen die folgenden Typen von Zeilen enthalten:

Leerzeilen
Leerzeilen werden ignoriert.

Kommentarzeilen
Ein Doppelkreuz (#) kann an beliebiger Stelle einer Zeile verwendet werden. **make** ignoriert alles, was dem Doppelkreuz auf derselben Zeile folgt.

Abhängigkeitszeilen
In Abhängigkeit von einem oder mehreren Zielen werden die folgenden Befehle ausgeführt. Die folgenden Schreibweisen sind möglich:

```
Ziele : Abhaengigkeiten
Ziele : Abhaengigkeiten ; Befehl
```

Die folgenden Befehle werden ausgeführt, wenn die *Abhängigkeiten*, deren Namen Platzhalterzeichen enthalten dürfen, neuer sind als die Ziele oder deren Ziele noch gar nicht existieren. Wenn

keine Abhängigkeiten angegeben werden, werden die Befehle immer ausgeführt, wenn das entsprechende Ziel angegeben wird. Vor den **Zielen** darf kein Tabulatorzeichen stehen.

Suffix-Regeln

Suffix-Regeln geben an, wie eine Datei mit dem zweiten Suffix aus einer Datei mit dem ersten Suffix erzeugt werden kann, wenn die Dateinamen vor dem Suffix identisch sind. Es kann eines der beiden folgenden Formate verwendet werden:

```
.Suffix1.Suffix2:
.Suffix:
```

Die zweite Schreibweise besagt, daß die Datei mit dem Basis-Dateinamen aus der Datei mit demselben Dateinamen und dem angehängten Suffix erzeugt werden kann.

Befehle

Befehle stehen unterhalb einer Abhängigkeitszeile. Befehlszeilen fangen grundsätzlich mit einem Tabulatorzeichen an. Wenn einer Befehlszeile ein Querstrich (–) vorangestellt wird, ignoriert **make** Fehler in der Befehlsausführung. Wenn der Befehlszeile ein Klammeraffe (@) vorangestellt wird, wird die Befehlszeile nicht auf dem Bildschirm ausgegeben, es sei denn, **make** wurde mit der Option **-n** aufgerufen.

Makro-Definitionen

Makro-Definitionen haben die folgende Form:

```
Name = String
     oder
define Name
String
endef
```

Die Leerzeichen vor und nach dem Gleichheitszeichen sind optional.

Include-Anweisungen

Include-Anweisungen ähneln den entsprechenden Anweisungen des C-Präprozessors und haben die Form:

```
include Dateien
```

Interne Makros

$? Die Liste der Abhängigkeiten, die neuer als das aktuelle Ziel sind. Kann nur in normalen Regeln, aber nicht in Suffix-Regeln verwendet werden.

$@ Der Name des aktuellen Ziels. Werden Bibliotheken erzeugt, bezeichnet dieses Makro den Namen der Bibliothek. Kann in normalen Regeln und Suffix-Regeln verwendet werden.

$< Der Name der aktuellen Abhängigkeit, die neuer ist als das aktuelle Ziel.

Programmieren

→

$* Der Name (ohne Suffix) der aktuellen Abhängigkeit, der neuer ist als das aktuelle Ziel. Kann nur in Suffix-Regeln verwendet werden.

$% Der Name der dazugehörigen .o-Datei, wenn das aktuelle Ziel eine Bibliothek ist. Kann in normalen Regeln und in Suffix-Regeln verwendet werden.

$^ Eine mit Leerzeichen getrennte Liste aller Abhängigkeiten; jede Abhängigkeit tritt nur einmal auf.

$+ Eine mit Leerzeichen getrennte Liste aller Abhängigkeiten; jede Abhängigkeit kann mehrfach auftreten.

Muster-Regeln

Muster-Regeln sind eine Generalisierung der Suffix-Regeln. Wenn sowohl Ziel als auch Abhängigkeit ein %-Zeichen enthalten, ersetzt **make** entsprechende Teile existierender Dateinamen. Beispielsweise kann die eingebaute Suffix-Regel

```
$(cc) -o $@ $<
```

folgendermaßen als Muster-Regel formuliert werden:

```
%.o : %.c
        $(cc) -o $@ $<
```

Makro-Modifikatoren

D Der Verzeichnis-Anteil jedes internen Makronamens mit Ausnahme von $?. Verwendungsmöglichkeiten:

$(*D) $(@D) $(?D) $(<D)
${%D) $(@D) $(^D)

F Der Datei-Anteil jedes internen Makronamens mit Ausnahme von $?. Verwendungsmöglichkeiten:

$(*F) $$(@F) $(?F) $(<F)
${%F} $(@F) $(^F)

Funktionen

$(subst *von, nach, String***)**
Ersetzen jedes Auftretens von *von* in *String* durch *nach*.

$(patsubst *Muster, nach, String***)**
Ähnlich wie **subst**, aber interpretiert % als Platzhalterzeichen in *Muster*. Ersetzt jedes Wort in *String*, das auf *Muster* paßt, durch *nach*.

$(strip *String***)**
Entfernen von überflüssigen Leerzeichen und Tabulatoren.

$(findstring *Substring, Hauptstring***)**
Gibt *Substring* zurück, wenn dieser in *Hauptstring* existiert, anderenfalls wird ein leerer String zurückgegeben.

$(filter *Muster, String***)**
Gibt diejenigen Wörter in *String* zurück, die auf mindestens ein Wort in *Muster* passen. *Muster* darf den Platzhalter % enthalten.

$(filter-out *Muster, String***)**
Entfernt diejenigen Wörter in *String*, die auf mindestens ein Wort in *Muster* passen. *Muster* darf das Platzhalterzeichen % enthalten.

$(sort *Liste***)**
Gibt *Liste* in lexikalisch sortierter Reihenfolge zurück.

$(dir *Liste***)**
Gibt den Verzeichnis-Anteil (alles bis zum letzten Schrägstrich) jedes Dateinamens in *Liste* zurück.

$(notdir *Liste***)**
Gibt den Dateinamen-Anteil (alles nach dem letzten Schrägstrich) jedes Dateinamens in *Liste* zurück.

$(suffix *Liste***)**
Gibt den Suffix-Anteil (alles nach dem letzten Punkt) jedes Dateinamens in *Liste* zurück.

$(basename *Liste***)**
Gibt den Basis-Dateinamen (alles bis zum letzten Punkt) jedes Dateinamens in *Liste* zurück.

$(addsuffix *Suffix,Liste***)**
Rückgabe jedes Dateinamens in *Liste*, wobei jeweils *Suffix* angehängt wird.

$(addprefix *Präfix,Liste***)**
Rückgabe jedes Dateinamens in *Liste*, wobei jeweils *Präfix* vorangestellt wird.

$(join *Liste1,Liste2***)**
Zurückgeben einer Liste, in der Wort für Wort die jeweiligen Elemente der beiden Listen aneinandergehängt werden. Beispiel: **`$(join a b,.c .o)`** ergibt **`a.c b.o`**.

$(word *n,String***)**
Gibt das *n*-te Wort von *String* zurück.

$(words *String***)**
Gibt die Anzahl der Wörter in *String* zurück.

$(firstword *Liste***)**
Gibt das erste Wort in **Liste** zurück.

$(wildcard *Muster***)**
Gibt eine Liste von Dateien im aktuellen Verzeichnis zurück, deren Namen auf *Muster* passen.

$(origin *Variable***)**
Gibt eine der folgenden Zeichenketten zurück, die angeben, wie die Variable definiert wurde: **undefined**, **default**, **environment**, **environment override**, **Datei**, **command line**, **override** oder **automatic**.

$(shell *Befehl***)**
Führt den *Befehl* aus und gibt das Ergebnis zurück. Zeilenwechsel in der Ausgabe des Befehls werden in Leerzeichen umgewandelt. Diese **make**-Funktion funktioniert ähnlich wie die Befehls-Substitution in den meisten Shells.

→

make
←

Ersetzen von Makro-Zeichenketten

${*Makro*:*s1*=*s2*}
: Evaluiert die aktuelle Definition von ${*Makro*}, nachdem für jedes Auftreten von *s1*, das direkt vor einem Leer- oder Tabulatorzeichen oder einem Zeilenende steht, *s2* eingesetzt wurde.

Besondere Ziele

.DEFAULT:
: Befehle, die zu diesem Ziel gehören, werden ausgeführt, wenn **make** keine Einträge oder Suffix-Regeln in der Beschreibungsdatei findet, mit denen das geforderte Ziel erzeugt werden könnte.

.EXPORT_ALL_VARIABLES:
: Wenn dieses Ziel in der Beschreibungsdatei angegeben wird, werden alle Makros zu den Kind-Prozessen exportiert.

.IGNORE:
: Fehlercodes von Befehlen werden ignoriert. Entspricht der -i Option.

.PHONY:
: Befehle, die zu einem so gekennzeichneten Ziel gehören, werden immer ausgeführt, auch wenn das Ziel existiert und aktuell ist.

.PRECIOUS:
: Dateien, die zu einem so gekennzeichneten Ziel gehören, werden nicht gelöscht, wenn **make** ein Signal (wie Interrupt) erhält, das die Bearbeitung abbricht, oder wenn der aufgerufene Befehl einen Fehlercode zurückgibt.

.SILENT:
: Die Befehle werden ausgeführt, aber nicht ausgegeben. Entspricht der -s Option.

.SUFFIXES:
: Die hier angegebenen Suffixe sollen in Suffix-Regeln berücksichtigt werden. Wenn keine Suffixe angegeben werden, werden die existierenden Suffix-Regeln abgeschaltet.

---

nm

nm [*Optionen*] [*objDateien*]

Ausgeben der Symboltabelle einer oder mehrerer Objektdateien in alphabetischer Reihenfolge. Wenn keine Dateien angegeben werden, wird die Symboltabelle der Datei *a.out* ausgegeben. Zur Ausgabe jedes Symbols gehören Wert, Typ, Größe, Name etc. Weiterhin kann ein Schlüsselbuchstabe, der das Symbol näher beschreibt, ausgegeben werden.

Optionen

-a, **--debug-syms**
: Ausgeben von Symbolen für den Debugger.

-f *Format*
: Angabe des Ausgabeformats (**bsd**, **sysv** oder **posix**). Der Default ist **bsd**.

-g, **--extern-only**
: Nur externe Symbole ausgeben.

-n, -v, **--numeric-sort**
: Sortieren der externen Symbole nach Adressen.

-p, --no-sort
Kein Sortieren der Symbole.

-r, --reverse-sort
Alphabetisches oder numerisches Sortieren der Symbole in umgekehrter Reihenfolge.

--size-sort Sortieren der Symbole nach Größe.

-u, --undefined-only
Nur undefinierte Symbole ausgeben.

-A, -o, -print-file-name
Ausgeben der Dateinamen, aus denen die jeweiligen Symbole stammen, vor den Symbolen.

-C, --demangle
Übersetzen von internen Symbolnamen in lesbare Versionen.

-D, --dynamic
Ausgeben dynamischer (nicht normaler) Symbole. Dies ist nur bei dynamischen Objekten (wie bspw. manchen dynamischen Bibliotheken) sinnvoll.

-P, --portability
Wie **-f posix**.

-V, --version
Ausgeben der Versionsnummer von **nm** auf Standardfehler.

patch [*Optionen*] [*Original* [*Patchdatei*]]

Anwenden der Patches in *Patchdatei* auf *Original*. Das Original wird durch die neue, gepatchte Version ersetzt. Eine Sicherungskopie des Originals wird nach *Original.orig* oder *Original~* geschrieben.

Optionen

+ [*Optionen*] [*Original2*]
Erneutes Anwenden der Patches mit anderen Optionen oder auf ein anderes Original.

-b *Suffix*, **--suffix=***Suffix*
Die Originaldatei wird in *Original.Suffix* gesichert.

-B *Präfix*, **--prefix=***Präfix*
Der Sicherung der Originaldatei wird *Präfix* vorangestellt.

-c, --context
Patchdatei wird als Kontext-Diff interpretiert.

-d *Verz*, **--directory=***Verz*
Wechseln nach *Verz*, bevor das Patchen beginnt.

-D *String*, **--ifdef=***String*
Alle Veränderungen am Original werden mit

```
#ifdef
String
#endif
```

gekennzeichnet.

→

patch
←

-e, --ed Der Inhalt von *Patchdatei* wird als **ed**-Befehle interpretiert.

-E, --remove-empty-files
: Wenn die Anwendung von **patch** zu leeren Dateien führt, werden diese gelöscht.

-f, --force Erzwingt alle Änderungen, auch wenn diese falsch aussehen. Wenn die Originaldatei nicht existiert, wird der entsprechende Patch übersprungen. Patches werden auch angewandt, wenn die Versionsnummer falsch ist. Patches werden als irreversibel angenommen.

-t, --batch Patches werden übersprungen, wenn die Originaldatei nicht existiert.

-F *num*, --fuzz=*num*
: Angeben der maximalen Anzahl von Zeilen, die ignoriert werden sollen, wenn **patch** entscheidet, an welcher Stelle ein Patch angebracht werden soll. Der Default ist 2. Diese Option ist nur bei Kontext-Diffs sinnvoll.

-l, --ignore-whitespace
: Bei Mustervergleichen werden Leer- und Tabulatorzeichen sowie Zeilensprünge ignoriert.

-n, --normal
: Interpretiert *Patchdatei* als normale Diff-Datei (kein Kontext-Diff).

-N, --forward
: Ignorieren von Patches, die entweder vermutlich rückgängig gemacht worden sind oder bereits angewendet wurden.

-o *Datei*, --output=*Datei*
: Schreiben der Ausgabe nach *Datei*.

-p[*num*], --strip[=*num*]
: Gibt an, wieviel vom Pfadnamen weggelassen wird. Wenn *num* 0 ist, wird alles weggelassen, so daß nur der Dateiname übrigbleibt. Wenn *num* 1 ist, wird der erste Schrägstrich weggelassen, jeder weitere Wert läßt einen weiteren Verzeichnisnamen weg.

-r *Datei*, --reject-file=*Datei*
: Schreibt nicht akzeptierte Abschnitte (Teile der Patchdatei, die nicht angewendet werden konnten) in die Datei *Datei*. Der Default ist *original.rej*.

-R, --reverse
: Ausführen eines umgedrehten Patch-Vorganges. Damit kann versucht werden, eine Patch-Operation rückgängig zu machen, indem die alten und neuen Dateien vertauscht werden.

-s, --silent, --quiet
: Unterdrücken der Ausgaben während des Patch-Vorganges.

-S, --skip Springen zum nächsten Patch in der Patch-Datei.

-u, --unified
: Interpretiert *Patchdatei* als vereinheitlichtes Kontext-Diff.

-V *Methode*, **--version-control=***Methode*
: Gibt an, wie Sicherungskopien erzeugt werden sollen (kann mit **-B** überschrieben werden). Erlaubte Werte für *Methode* sind:

  t, numbered
  : Numerierte Sicherungskopien.

  nil, existing
  : Dateien werden anhand bereits vorhandener Sicherungen gesichert. Einfache Sicherungen sind der Default. Dies ist das Defaultverhalten von **patch**.

  never, simple
  : Es werden nur einfache Sicherungskopien angelegt.

TMPDIR
: Gibt das Verzeichnis für temporäre Dateien an, der Default ist */tmp*.

SIMPLE_BACKUP_SUFFIX
: Enthält ein Suffix, das an Sicherungskopien anstelle von *.orig* oder ~ angehängt wird.

VERSION_CONTROL
: Gibt an, wie Sicherungskopien benannt werden sollen (siehe **-V**).

rpcgen [*Optionen*] *Datei*

rpcgen parst die in RPC geschriebene Datei *Datei* und erzeugt ein C-Programm, das die RPC-Beschreibung implementiert. Header-Dateien, die aus *Datei*.**x** erzeugt wurden, werden nach *Datei*.**h**, XDR-Routinen nach *Datei*_**xdr.c**, Server-Code nach *Datei*_**svc.c** und Client-Code nach *Datei*_**clnt.c** geschrieben. Zeilen, die mit einem %-Zeichen beginnen, werden ignoriert. Defaultmäßig erzeugt **rpcgen** Code, der kompatibel ist mit SunOS 4.1. Nähere Informationen zu RPC finden Sie in *Power Programming with RPC* von John Bloomer.

-a
: Erzeugen aller Dateien (Client- und Server-Code).

-5
: Erzeugen von SVR4-kompatiblem Code.

-c
: Erzeugen von XDR-Routinen. Kann nicht in Verbindung mit anderen Optionen verwendet werden.

-C
: Erzeugen von ANSI-C- Code (Default).

-k
: Erzeugen von K&R-C-Code.

-D*Name*[=*Wert*]
: Definieren des Symbols *Name*. Der Wert von *Name* ist *Wert* oder 1, falls *Wert* nicht angegeben wird.

-h
: Ausgeben einer Header-Datei. Wenn auch -T angegeben wird, enthält die Header-Datei Code für RPC-Sprungtabellen. Kann nicht in Verbindung mit anderen Optionen verwendet werden.

→

rpcgen ←

- **-I** — Produziert einen Server, der mit **inetd** kompatibel ist.
- **-K** *Sek* — Gibt die Zeit an, die der Server warten soll, nachdem er eine Anfrage beantwortet hat und bevor er sich beendet. Der Default ist 120. Wenn *Sek* -1 ist, beendet sich der Server nie.
- **-l** — Erzeugt Client-Code. Kann nicht in Verbindung mit anderen Optionen verwendet werden.
- **-m** — Nur Server-Code erzeugen. Es wird keine main()-Funktion erzeugt. Kann nicht in Verbindung mit anderen Optionen verwendet werden.
- **-N** — Neuer Stil. Prozeduren dürfen mehr als ein Argument haben. Nicht unbedingt kompatibel mit dem alten Stil.
- **-o** [*Datei*] — Schreiben der Ausgabe nach *Datei* oder nach Standardausgabe.
- **-Ss** — Nur ein Server-Code-Gerüst erzeugen.
- **-t** — Erzeugen einer RPC-Sprungtabelle. Kann nicht in Verbindung mit anderen Optionen verwendet werden.
- **-T** — Unterstützung für RPC-Sprungtabellen erzeugen.

size

size [*Optionen*] [*Objdatei . . .*]

Ausgeben der Anzahl an Bytes in jeder Sektion von *Objdatei* sowie der Gesamtgröße. Wenn *Objdatei* nicht angegeben wird, wird *a.out* angenommen.

- **-d** — Ausgeben der Größe in dezimaler und hexadezimaler Darstellung.
- **--format** *Format* — Formatieren der Ausgabe im System V-Format (**--format sysv**) oder BSD-Format (**--format berkeley**).
- **-o** — Ausgeben der Größe in oktaler und hexadezimaler Darstellung.
- **--radix** *Num* — Gibt an, wie die Größe ausgegeben werden soll: hexadezimal und dezimal (wenn *Num* 10 oder 16 ist) oder hexadezimal und oktal (wenn *Num* 8 ist).
- **-x** — Gibt die Größe in hexadezimaler und dezimaler Darstellung an.
- **-A** — Verhalten entsprechend dem System-V-**size**-Befehl.
- **-B** — Verhalten entsprechend dem BSD-**size**-Befehl.

strip

strip [*Optionen*] *Dateien*

Entfernen der Symbole aus den Objektdateien *Dateien*, wodurch die Größe der Objektdateien reduziert und so Plattenplatz gespart wird.

Optionen

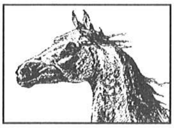
- -F *format*, **--target=***Format*
  Nimmt an, daß die Eingabedateien im Format *Format* sind.
- -O *Format*, **--output-target=***Format*
  Schreibt die Ausgabedateien im Format *Format*.
- -R *section*, **--remove-section=***Sektion*
  Löscht die Sektion *Sektion*.
- -s, **--strip-all**
  Entfernt alle Symbole.
- -S, -g, **--strip-debug**
  Entfernt Debugging-Symbole.
- -x, **--discard-all**
  Entfernt nicht-globale Symbole.
- -X, **--discard-locals**
  Entfernt vom Compiler erzeugte lokale Symbole.
- -v, **--verbose**
  „Verbose-Modus".

yacc [*Optionen*] *Datei*

Erzeugt aus einer Eingabedatei, die eine kontextfreie Grammatik enthält, Parsing-Tabellen und schreibt diese nach *y.tab.c*. Der Name **yacc** steht für „yet another compiler-compiler". Siehe auch **flex**, **bison** und *lex & yacc* von John Levine, Tony Mason und Doug Brown.

Optionen

- -b *Präfix* Stellt dem Namen der Ausgabedatei *Präfix* anstelle von *y* voran.
- -d Erzeugt eine Datei *y.tab.h*, die **#define**-Anweisungen enthält, welche die Token von **yacc** auf die vom Benutzer deklarierten Tokennamen abbilden.
- -l Unterdrücken der **#line**-Anweisungen in *y.tab.c*. Diese Option sollte nur verwendet werden, wenn die Parsing-Tabellen keinen Fehler mehr enthalten.
- -t Erzeugen von Laufzeit-Debugging-Code.
- -v Erzeugen von *y.output*, einer Datei, die diagnostische Informationen und Anmerkungen zu den Parsing-Tabellen enthält.



KAPITEL 13

Überblick über die System- und Netzwerkverwaltung

Häufig benutzte Befehle

Im folgenden finden Sie Tabellen mit den wichtigsten Befehlen zur Systemverwaltung.

Archivieren

cpio	Dateiarchive anlegen, kopieren und auslesen.
tar	Sichern oder Zurückholen von Dateien in und aus Archivmedien oder anderen Dateien.

Uhrzeit

clock	Verwalten der CMOS-Uhr.
netdate	Uhrzeit anhand eines anderen Rechners einstellen.
rdate	Uhrzeit-Server verwalten.
zdump	Liste der Zeitzonen ausgeben.
zic	Dateien zur Zeitumrechnung anlegen.

Dämonen

bootpd	Internet-Boot-Protocol-Dämon.
fingerd	Finger-Dämon.
ftpd	File-Transfer-Protocol-Dämon.
identd	Herausfinden des Benutzers, der einen TCP/IP-Prozeß gestartet hat.
imapd	IMAP-Protocol-Mailbox-Server-Dämon.
inetd	Internet-Superserver-Dämon.
kerneld	Automatisches Laden von Kernel-Modulen.
klogd	Verwalten des syslogd.
lpd	Drucker-Dämon.
mountd	NFS-Server-Dämon.
named	Internet Domain Name Server.

nfsd	NFS-Client-Dämon.
pop2d	POP2 Server.
pop3d	POP3 Server.
powerd	Überwacht die Verbindung zu unterbrechungsfreien Stromversorgungen.
pppd	Dämon für das Point-to-Point Protocol (PPP).
rdistd	Server für die Verteilung von Software.
rexecd	Server für die entfernte Ausführung von Programmen.
rlogind	Server für das rlogin-Kommando.
routed	Routing-Dämon.
rwhod	Server für entfernte who-Befehle.
syslogd	Dämon, der Systemmeldungen protokolliert.
tcpd	TCP-Netzwerk-Dämon.
tftpd	Dämon für das Trivial File Transfer Protocol (TFTP).
ypbind	NIS-Binder-Prozeß
yppasswdd	NIS-Server für die Änderung von Paßwörtern.
ypserv	NIS-Server-Prozeß.

Hardware

arp	Verwalten des ARP-Caches.
cfdisk	Verwalten von Festplattenpartitionen.
fdisk	Verwalten von Festplattenpartitionen.
kbdrate	Einstellen der Tastenwiederholrate der Tastatur.
ramsize	Ausgeben von Informationen über die RAM-Disk.
setserial	Einstellen der seriellen Schnittstellen.
slattach	Serielle Leitungen als Netzwerkschnittstellen ansprechen.

Rechnerinformation

dnsdomainname	Ausgeben des DNS-Domänennamens.
domainname	Ausgeben des NIS-Domänennamens.
host	Ausgeben von Rechner- und Zoneninformation.

Installation

install	Kopieren von Dateien an Stellen, an denen Benutzer auf sie zugreifen können.
rdist	Dateien an andere Systeme weitergeben.
cpio	Dateiarchive anlegen, kopieren und auslesen.
tar	Sichern oder Zurückholen von Dateien in und aus Archivmedien oder anderen Dateien.

Email

makemap	Datenbankdateien von **sendmail** aktualisieren.
rmail	**uucp**-Email weiterleiten.
sendmail	Eingehende und ausgehende Email weiterleiten.

Verwalten des Kernels

insmod	Laden eines neuen Kernelmoduls.
lsmod	Anzeigen der geladenen Kernelmodule.
modprobe	Laden von Kernelmodulen unter Berücksichtigung aller Abhängigkeiten.
rmmod	Entladen eines neuen Kernelmoduls.

Verwalten von Dateisystemen

Auf UNIX-Systemen ist ein *Dateisystem* Hardware, die formatiert worden ist, um darauf Dateien ablegen zu können. Dateisysteme kann es auf Festplatten, Disketten, CD-ROMs oder anderen Speichermedien geben, auf die wahlfrei zugegriffen werden kann.

Das genaue Format und Verfahren, mit denen die Dateien abgelegt werden, sind hier unwichtig, weil das System eine einheitliche Schnittstelle für alle ihm bekannten *Dateisystemtypen* bereitstellt. Unter Linux ist das bekannteste Dateisystem das „Second Extended Filesystem" *(ext2fs)*, das vermutlich auch Sie zum Abspeichern von Dateien verwenden. Das Second Extended Filesystem wurde hauptsächlich für Linux entwickelt und unterstützt Dateinamen mit bis zu 256 Zeichen, eine maximale Dateisystemgröße von vier Terabyte und andere nützliche Dinge. (Es heißt „second", also „zweites", weil es der Nachfolger des ersten Dateisystems, des „Extended Filesystems", ist.) Andere gebräuchliche Dateisystemtypen sind das MS-DOS-Dateisystem, mit dem unter Linux auf MS-DOS-Partitionen und -Disketten zugegriffen werden kann, und das ISO 9660-Dateisystem, das von CD-ROMs verwendet werden kann.

debugfs	Fehlersuche im **extfs**-Dateisystem.
dumpe2fs	Ausgeben von Informationen über den Superblock und Blockgruppen.
e2fsck	Überprüfen und Reparieren eines Second Extended Dateisystems.
fsck	Überprüfen und Reparieren von Dateisystemen.
fsck.minix	Überprüfen und Reparieren eines MINIX-Dateisystems.
fuser	Ausgeben der Prozesse, die ein Dateisystem benutzen.
mkfs	Anlegen eines neuen Dateisystems.
mkfs.ext2	Anlegen eines neuen Second Extended Dateisystems.
mkfs.minix	Anlegen eines neuen MINIX-Dateisystems.
mklost+found	Anlegen des *lost+found*-Verzeichnisses.
mkswap	Anlegen eines Auslagerungsbereiches.
mount	Einhängen eines Dateisystems.
rdev	Parameter des Wurzeldateisystems im Kernelimage ändern.
rootflags	Ausgeben oder Setzen der Flags, mit denen das Wurzel-Dateisystem eingehängt wurde.
showmount	Ausgeben, an welche Rechner Dateisysteme exportiert werden.
swapdev	Anzeigen oder Einstellen von Parametern des Auslagerungsbereiches.
swapon	Verwenden einer Partition oder Datei zum Auslagern.
swapoff	Partition oder Datei nicht mehr zum Auslagern verwenden.
sync	Buffer der Dateisysteme auf die Medien schreiben.
tune2fs	Einstellen eines Second Extended Dateisystems.
umount	Aushängen eines Dateisystems.

Verschiedenes

cron	Befehle zur Ausführung zu bestimmten Zeiten vormerken.
dmesg	Anzeigen der Startmeldungen, nachdem das System gestartet worden ist.
install	Kopieren von Dateien und Einstellen der Zugriffsrechte.
ldconfig	Aktualisieren der Links auf Bibliotheken und des Caches.
logger	Mitteilungen an das Systemprotokoll schicken.
login	Anmelden.
rdist	Dateien zwischen Rechnern transportieren.
rstat	Anzeigen des Systemstatus von Rechnern.
run-parts	Alle Skripte in einem Verzeichnis ausführen.
script	Protokoll der eingegebenen Befehle und ihrer Ausgaben erstellen.

Netzwerke

gated	Verwalten von Routing-Tabellen zwischen Netzwerken.
ifconfig	Verwalten von Netzwerkschnittstellen.
ipfwadm	Firewall- und Kontoführungsfunktionen verwalten.
named	Übersetzen zwischen Domänennamen und IP-Adressen.
netstat	Ausgeben des Zustandes des Netzwerks.
portmap	Abbilden von Dämonen auf Ports.
rarp	Verwalten der RARP-Tabelle.
route	Verwalten von Routing-Tabellen.
routed	Routing-Tabellen dynamisch aktualisieren.
rpcinfo	RPC-Information ausgeben.
traceroute	Verfolgen des Weges von Paketen zu einem entfernten Rechner.

Verwaltung von NIS

domainname	Einstellen oder Anzeigen der aktuellen NIS-Domäne.
makedbm	NIS-Datenbanken neu aufbauen.
ypbind	Mit einem NIS-Server verbinden.
ypcat	Inhalt der NIS-Datenbank ausgeben.
ypinit	Neue NIS-Datenbank anlegen.
ypmatch	Ausgeben des Wertes eines oder mehrerer NIS-*Schlüssel*.
yppasswd	Ändern des Paßwortes eines Benutzers in der NIS-Datenbank.
yppasswdd	Aktualisieren der NIS-Datenbank nach **yppasswd**.
yppoll	Bestimmen der Version einer NIS-Tabelle im NIS-Server.
yppush	NIS-Tabelle weiterverbreiten.
ypserv	NIS-Server-Dämon.
ypset	**ypbind** auf einen bestimmten Server einstellen.
ypwhich	Anzeigen des Namens des NIS-Servers oder des primären Tabellenlieferanten.
ypxfr	Übertragen einer NIS-Datenbank vom Server auf den lokalen Rechner.

Drucken

tunelp	Einstellen der Drucker-Parameter.

Sicherheit und Systemintegrität

badblocks	Suchen nach fehlerhaften Blöcken.
chroot	Ändern des Wurzelverzeichnisses.

Starten und Anhalten des Systems

bootpd	Internet-Boot-Protocol-Dämon.
bootpgw	Internet-Boot-Protocol-Gateway.
bootptest	Test des **bootpd**.
halt	System anhalten oder herunterfahren.
reboot	System herunterfahren und neu starten.
runlevel	Ausgeben des Runlevels.
shutdown	System herunterfahren.
telinit	Ändern des aktuellen Runlevels.
uptime	Anzeigen der Zeit, die seit dem Systemstart vergangen ist.

Systemaktivität und Prozeßverwaltung

Eine Reihe der Befehle in Kapitel 1, *Linux-Benutzerbefehle*, sind besonders hilfreich bei der Verwaltung von Prozessen. Dazu gehören z.B. **kill**, **killall**, **killall5**, **pidof**, **ps** und **who**.

fuser	Identifizieren von Prozessen, die eine Datei oder ein Dateisystem verwenden.
netstat	Anzeigen des Netzwerkzustands.
psupdate	Aktualisieren der Datei */boot/psupdate*.

Benutzer

adduser	Hinzufügen eines neuen Benutzers.
rusers	Ausgeben von **who**-ähnlichen Informationen über andere Rechner.
rwall	Ausgeben einer Meldung an Benutzer auf entfernten Rechnern.
w	Anzeigen der angemeldeten Benutzer.
wall	Ausgeben einer Meldung an alle angemeldeten Benutzer.

Netzwerk-Überblick

Netzwerke verbinden Computer, so daß verschiedene Systeme untereinander Informationen austauschen können. UNIX-Systeme stellen Benutzern wie Systemverwaltern traditionell einfache, aber mächtige Netzwerkdienste zur Verfügung, mit denen Sie überprüfen können, ob andere Rechner eingeschaltet und erreichbar sind, auf Dateien von anderen Systemen zugreifen können, miteinander via Email kommunizieren können und vieles andere mehr.

Bei den meisten Netzwerkbefehlen muß auf jedem System ständig ein sogenannter Server-Prozeß im Hintergrund laufen, der auf Benutzeranfragen wartet und diese dann abarbeitet. Solche Prozesse nennt man auch *Dämonen*. Typische Beispiele, die grundlegende Funktionalitäten des Linux-Systems sicherstellen, sind **named** (übersetzt numerische IP-Adressen in alphanumerische Rechnernamen, die den Menschen so gut gefallen), **lpd** (sorgt für das Spooling) und **ftpd** (ermöglicht es Ihnen, sich via **ftp** zu einer anderen Maschine zu verbinden).

Die meisten UNIX-Netzwerkbefehle basieren auf den Internet-Protokollen. Bei diesen handelt es sich um standardisierte Methoden zur Kommunikation in einem Netzwerk über verschiedene, hierarchisch angeordnete Schichten. In den unteren Schichten behandeln die Protokolle z.B. Adressierung und Versandwege, in den oberen Schichten werden z.B. Benutzer lokalisiert und Befehle der Benutzer ausgeführt.

Die grundlegenden Benutzerbefehle, die bei den meisten Systemen, welche die Internet-Protokolle verwenden, vorhanden sind, werden üblicherweise nach den beiden gebräuchlichsten Protokollen TCP/IP-Befehle genannt. Sie können diese Befehle benutzen, um mit anderen UNIX-Systemen zu kommunizieren, auch solchen, auf denen nicht Linux läuft. Viele der Befehle können auch zur Kommunikation mit Systemen verwendet werden, auf denen kein UNIX läuft, weil heutzutage sehr viele Systeme TCP/IP oder einen Teil davon unterstützen.

In diesem Kapitel werden auch NFS und NIS behandelt, mit denen Dateien und andere Informationen transparent in einem Netzwerk verteilt werden können, sowie **sendmail**.

Befehle zur Verwaltung von TCP/IP

ftpd	Server für Dateiübertragungen.
gated	Verwalten von Routing-Tabellen zwischen Netzwerken.
named	Übersetzen von Domänen- und Rechnernamen in IP-Adressen.
netstat	Ausgeben des Netzwerkzustands.
ifconfig	Einstellen der Parameter der Netzwerkschnittstelle.
nslookup	Anfragen an DNS-Server stellen.
ping	Überprüfen, ob ein entfernter Rechner erreichbar ist und antwortet.
pppd	Aufbauen von seriellen Verbindungen mittels PPP.
rdate	Absenden einer Mitteilung an den Time Server, daß sich das Datum geändert hat.
route	Verwalten von Routing-Tabellen.
routed	Dynamisches Aktualisieren von Routing-Tabellen.
slattach	Serielle Leitung als Netzwerkschnittstelle einrichten.
telnetd	Ermöglichen von TELNET-Verbindungen zu diesem Rechner.
tftpd	Server für das TFTP-Protokoll.

Befehle zur Verwaltung von NFS und NIS

portmap	Abbilden von Dämonen auf Ports.
rpcinfo	RPC-Information ausgeben.
domainname	Einstellen oder Anzeigen der aktuellen NIS-Domäne.
makedbm	NIS-Datenbanken neu aufbauen.
ypbind	Mit einem NIS-Server verbinden.
ypcat	Inhalt der NIS-Datenbank ausgeben.
ypinit	Neue NIS-Datenbank anlegen.
ypmatch	Ausgeben des Wertes eines oder mehrerer NIS-*Schlüssel*.
yppasswd	Ändern des Paßwortes eines Benutzers in der NIS-Datenbank.
yppasswdd	Aktualisieren der NIS-Datenbank nach **yppasswd**.
yppoll	Bestimmen der Version einer NIS-Tabelle im NIS-Server.
yppush	NIS-Tabelle weiterverbreiten.
ypserv	NIS-Server-Dämon.
ypset	**ypbind** auf einen bestimmten Server einstellen.
ypwhich	Anzeigen des Namens des NIS-Servers oder primären Tabellenlieferanten.
ypxfr	Übertragen einer NIS-Datenbank vom Server auf den lokalen Rechner.

Überblick über TCP/IP

TCP/IP ist eine Gruppe von Protokollen zur Kommunikation, die definieren, wie sich verschiedene Computer untereinander verständigen können. Der Name stammt von den beiden wichtigsten Protokollen, dem Transmission Control Protocol und dem Internet Protocol. Das Internet Protocol beschreibt, wie Daten in Form von Paketen zwischen Rechnern transportiert werden: Die Daten werden in Pakete aufgeteilt, die über andere Rechner im Netzwerk an ihr Ziel transportiert werden. Das Transmission Control Protocol sorgt dafür, daß die Daten einer Nachricht am Ziel in der richtigen Reihenfolge wieder zusammengesetzt werden und daß eventuell fehlende Daten noch einmal gesendet werden, bis sie korrekt empfangen werden konnten.

Weitere Protokolle der TCP/IP-Familie:

Address Resolution Protocol (ARP)
: Übersetzung zwischen Internet- und lokalen Hardware-Adressen (z.B. Ethernet).

Internet Control Message Protocol (ICMP)
: Kontroll- und Fehlermeldungsprotokoll.

Point-to-Point Protocol (PPP)
: Stellt synchrone und asynchrone Netzwerkverbindungen über serielle Leitungen bereit.

Reverse Address Resolution Protocol (RARP)
: Übersetzung zwischen lokalen Hardware- und Internet-Adressen (Gegenstück zu ARP).

Serial Line Internet Protocol (SLIP)
: IP über serielle Leitungen.

Simple Mail Transport Protocol (SMTP)
: Wird von **sendmail** verwendet, um Email via TCP/IP zu versenden.

Simple Network Management Protocol (SNMP)
: Verwalten verteilter Netzwerke über TCP/IP.

User Datagram Protocol (UDP)
: Datenübertragung ohne die Verläßlichkeit von TCP.

Hintergrundinformationen zu TCP/IP finden Sie in dem dreibändigen Werk *Internetworking with TCP/IP* von Douglas R. Comer, erschienen bei Prentice-Hall, sowie in dem ebenfalls dreibändigen Werk *TCP/IP Illustrated* von W. Richard Stevens, erschienen bei Addison-Wesley. Genauere Beschreibungen der Befehle in diesem und im nächsten Kapitel finden Sie in *TCP/IP Netzwerk Administration* von Craig Hunt, erschienen im O'Reilly Verlag.

In der Architektur der TCP/IP-Protokolle werden die Daten durch einen Stack (in Richtung auf die sogenannte *Network Access Layer*, die Netzwerkzugriffsschicht) geschickt und von dort über das Netz übertragen. Am anderen Ende durchlaufen die Daten den Stack in entgegengesetzter Richtung (siehe Abbildung 13-1).

IP-Adressen

Die IP-Adresse (also quasi die „Internet-Adresse“) eines Rechners ist eine 32 Bit große Zahl, die einen Rechner eindeutig identifiziert. Jeder Rechner muß eine eindeutige IP-Adresse haben. Eine IP-Adresse besteht aus zwei Teilen: dem Netzwerkteil und dem Rechnerteil. Die Anzahl der Bits in der Adresse, die das Netzwerk und den Rechner identifizieren, ist abhängig von der Klasse der Adresse. Es gibt im wesentlichen drei Adreßklassen: A, B und C (siehe Abbildung 13-2). Die am weitesten links stehenden Bits geben an, zu welcher Klasse eine Adresse gehört.

Abbildung 13-1: Schichten in der Architektur des TCP/IP-Protokolls

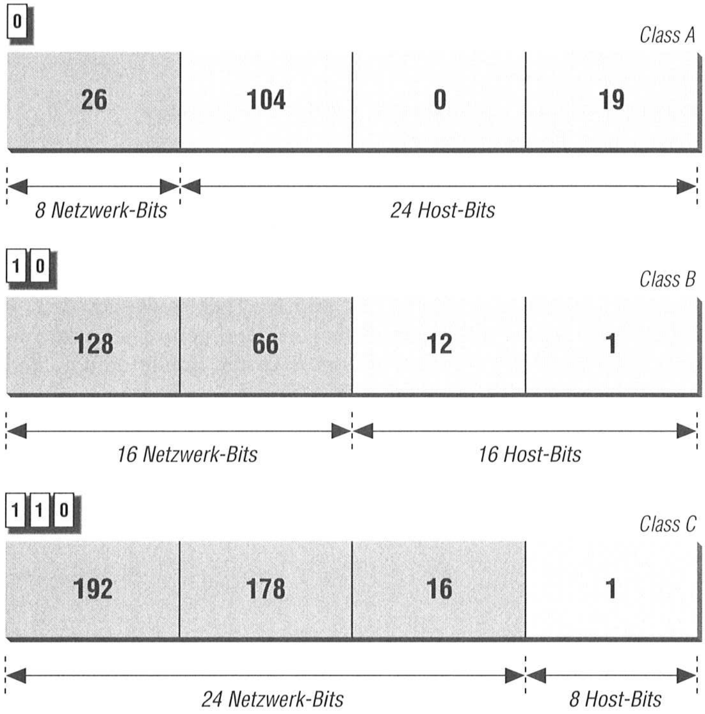


Abbildung 13-2: IP-Adreßstruktur

Wenn Sie Ihren Rechner in das Internet bringen möchten, wenden Sie sich an das für Sie zuständige Network Information Center oder Ihren Provider, die Ihnen eine IP-Adresse zuteilen werden. Wenn Sie sich nicht mit anderen Netzwerken verbinden wollen, können Sie Ihre IP-Adresse beliebig wählen, solange Sie sich an die Syntax für IP-Adressen halten. Sie sollten dann eine der reservierten Adressen für private Netzwerke verwenden, die im RFC 1597 angegeben sind und nicht bei der IANA (Internet Assigned Numbers Authority, einem Büro, das international im Internet verwendete Nummern wie IP-Adressen und RFC-Nummern verwaltet) registriert werden müssen. Eine IP-Adresse ist etwas anderes als eine Ethernet-Adresse, die vom Hersteller der Ethernet-Karte festgelegt wird.

Gateways und Routing

Gateways sind Rechner, die Routing-Informationen und Datenpakete zwischen verschiedenen Netzwerken vermitteln. Jeder Teil eines Netzwerks, der einer separaten lokalen Verwaltung untersteht, wird als autonomes System (AS) bezeichnet. Autonome Systeme werden untereinander über externe Gateways verbunden. Ein autonomes System kann selbst wiederum ein System von Netzwerken enthalten, die über interne Gateways miteinander verknüpft sind.

Gateway-Protokolle

Zu den Gateway-Protokollen gehören:

EGP (Exterior Gateway Protocol)

BGP (Border Gateway Protocol)
: Protokolle zum Informationsaustausch für externe Gateways.

RIP (Routing Information Protocol)
: Internes Gateway-Protokoll, populär in LANs.

Hello Protocol
: Internes Gateway-Protokoll.

Routing-Dämonen

gated und **routed**, die Routing-Dämonen, versetzen einen Rechner in die Lage, als Gateway fungieren zu können. Nur einer der beiden kann zur Zeit auf einem Rechner laufen. **gated**, der Gateway Routing Daemon, ermöglicht es einem Rechner, sowohl als externes als auch als internes Gateway zu arbeiten. Die Konfiguration wird dadurch erleichtert, daß die Protokolle RIP, Hello, BGP und EGP in einem Paket kombiniert sind.

routed, der Netzwerk-Routing-Dämon, ermöglicht nur die Funktion eines internen Gateways. Er verwaltet die Internet- Routing-Tabellen. Mehr Information zu **gated** und **routed** finden Sie bei den TCP/IP-Befehlen in Kapitel 14, *Befehle zur System- und Netzverwaltung*.

Routing-Tabellen

Routing-Tabellen enthalten die Informationen, die benötigt werden, um ein Paket an sein Ziel zu leiten. Zu diesen Informationen gehören u.a. das Zielnetzwerk, der zu verwendende Gateway, der Zustand der Route und die Anzahl der übertragenen Pakete. Routing-Tabellen können mit dem **netstat**-Befehl angezeigt werden.

Name Service

Jeder Rechner in einem Netzwerk hat einen Namen, der Informationen über ihn enthält. Solche Rechnernamen können jedem Gerät, das über eine IP-Adresse verfügt, zugewiesen werden. Namen-Dienste übersetzen Rechnernamen, die für Menschen leicht zu merken sind, in IP-Adressen, mit denen Computer besser arbeiten können.

DNS und BIND

Der Domain Name Service (DNS) ist eine verteilte Datenbank von Informationen über Rechner eines Netzwerks. Die Struktur von DNS ist der der UNIX-Dateisysteme sehr ähnlich: ein umgedrehter Baum mit der Wurzel an der Spitze. Die bekannteste Implementation von DNS ist BIND (Berkeley Internet Name Domain).

DNS arbeitet nach dem Client-Server-Modell. Ein sogenannter *Resolver* ist der Client, ein Programm, das Anfragen über Rechner stellt. Der *Name Server* ist ein Prozeß, der diese Anfragen beantwortet. Der Server im BIND-Paket ist der Dämon **named**. Name Server können mit dem **nslookup**-Befehl interaktiv nach Informationen über Rechner befragt werden. Weitere Details zu **named** und **nslookup** finden Sie bei den TCP/IP-Befehlen in Kapitel 14 sowie in *DNS und BIND* von Paul Albitz und Cricket Liu.

Wenn Ihr Rechner der Name Server seiner Domäne ist, ist der Rechner dafür verantwortlich, die Namen der Rechner seiner Domäne zu verwalten und auf Anfrage herauszugeben. Andere Name Server im Internet werden ihn danach fragen.

Domänennamen

Der vollständige Domänenname ist eine Folge von Namen, die mit der eigenen Domäne beginnt und weiter zur Wurzeldomäne geht, wobei die Namen jeweils durch Punkte getrennt sind. Beispielsweise bezeichnet *oreilly.de* die Domäne *oreilly* (des O'Reilly Verlages), die unterhalb der erstrangigen Domäne *de* (für Deutschland) eingeordnet ist. Ein Rechner in dieser Domäne ist beispielsweise *www.oreilly.de*. Amerikanische Domänen sind traditionell in den folgenden erstrangigen Domänen zu finden, obwohl auch Domänen anderer Länder sich hier einordnen dürfen und es auch eine spezielle nationale Domäne für die USA (*.us)* gibt:

com	Kommerzielle Unternehmen
edu	Bildungseinrichtungen
gov	Staatliche Einrichtungen
mil	Militärische Einrichtungen
net	Kommerzielle Internet-Organisationen, oft Internet Service Provider.
org	Diverse Organisationen.

In den meisten anderen Ländern werden länderspezifische erstrangige Domänen wie *de* für Deutschland, *at* für Österreich und *ch* für die Schweiz verwendet.

Registrieren von Domänen

Wenn Sie eine Domäne einrichten wollen, die öffentlich erreichbar sein wird, müssen Sie Ihre Domäne registrieren lassen. Genaue Informationen dazu finden Sie unter *http://www.internic.net/* und *http://www.nic.de/*.

Konfigurieren von TCP/IP

ifconfig

Die Netzwerkschnittstelle gibt an, wie die Netzwerksoftware die Hardware benutzt: Treiber, IP-Adresse usw. Um eine Netzwerkschnittstelle zu konfigurieren, verwenden Sie den **ifconfig**-Befehl. Mit **ifconfig** weisen Sie während der Startphase des Rechners einer Netzwerkschnittstelle eine IP-Adresse zu und setzen die Netzwerkmaske und Broadcast-Adresse. Sie können auch weitere Parameter der Schnittstelle einstellen, z.B. die Verwendung von ARP, den Einsatz von treiber-spezifischem Code zur Fehlersuche oder die Adresse der Gegenstelle in einer Point-to-Point-Verbindung. Mehr Information zu **ifconfig** finden Sie bei den TCP/IP-Befehlen in Kapitel 14.

Kommunikation über serielle Leitungen

Es gibt zwei Protokolle für die Kommunikation über serielle Leitungen: Serial Line IP (SLIP) und Point-to-Point Protocol (PPP). Mit diesen Protokollen können Computer Daten über eine serielle Schnittstelle und ein serielles Kabel anstatt über eine Netzwerkkarte und ein Ethernet-Kabel übertragen.

Unter Linux wird der Treiber für das SLIP-Protokoll im Kernel installiert. Um eine serielle Leitung auf SLIP einzustellen, wird das **slattach**-Programm verwendet (nähere Informationen zu **slattach** in Kapitel 14). Vergessen Sie bitte nicht, daß Sie auch nach dem Aufruf von **slattach** immer noch mit **ifconfig** die Netzwerkschnittstelle konfigurieren müssen. Wenn Ihr Rechner beispielsweise *dilbert* heißt und Sie sich in den Rechner *dogbert* einwählen wollen, geben Sie ein:

```
# ifconfig sl0 dilbert pointopoint dogbert
# route add dogbert
# route add default gw dogbert
```

Damit wird die Schnittstelle als Point-to-Point-Verbindung zu **dogbert** konfiguriert und eine Route zu **dogbert** eingerichtet und als Default-Route festgelegt, womit **dogbert** als Gateway angegeben wird.

PPP wurde entwickelt, um einige der Probleme mit SLIP zu beseitigen: PPP kann auch Pakete aus Nicht-Internet-Protokollen bearbeiten und bietet Autorisierung der Clients und Van Jacobson-Headerkomprimierung. Außerdem kann PPP jedes Protokoll, das durch PPP hindurch geleitet wird, dynamisch konfigurieren. Unter Linux besteht PPP zum einen als Treiber im Kernel und zum anderen als Dämon **pppd**. Weitere Informationen zu **pppd** in Kapitel 14.

Problemsuche mit TCP/IP

Mit den folgenden Befehlen können TCP/IP-Probleme gefunden und behoben werden. Weitere Details zu diesen Befehlen finden Sie bei den TCP/IP-Befehlen in Kapitel 14.

ifconfig	Ausgeben von Informationen über die grundlegende Konfiguration einer Netzwerkschnittstelle.
netstat	Ausgeben des Netzwerkzustands.
ping	Ausgeben, ob ein entfernter Rechner erreicht werden kann.
nslookup	Abfragen des DNS Name Services.
traceroute	Verfolgen der Route eines Pakets auf dem Weg zu einem entfernten Rechner.

Überblick über NFS

NFS ist ein verteiltes Dateisystem, mit dem Benutzer entfernte Dateisysteme einhängen, als ob sie auf dem lokalen Rechner vorhanden wären. NFS verwendet ein Client-Server-Modell, in dem ein Server Verzeichnisse exportiert und Clients diese Verzeichnisse in ihren Verzeichnisbaum einhängen, um an die in ihnen enthaltenen Dateien heranzukommen. Mit NFS ist es nicht mehr nötig, Kopien von Dateien auf mehreren Rechnern zu halten, weil alle Clients auf eine einzige Kopie auf dem Server zugreifen können. NFS ist ein RPC-basiertes Protokoll der Applikationsschicht. Weitere Informationen zu Netzwerkprotokollen finden Sie im Abschnitt *Überblick über TCP/IP*.

Verwalten von NFS

Um NFS-Clients und -Server einzurichten, müssen die NFS-Dämonen gestartet werden, der NFS-Server muß die Dateisysteme exportieren, und die Clients müssen sie bei sich einhängen. Die Datei */etc/exports* ist die Konfigurationsdatei des Servers; sie gibt an, welche Dateien und Verzeichnisse exportiert werden und wer wie darauf zugreifen darf. Die Namen und Adressen von Rechnern, die zugreifen dürfen, sind in der */etc/hosts*-Datei abgelegt.

Dämonen

Der Server-Dämon von NFS heißt **nfsd** und nimmt RPC-Aufrufe von Clients entgegen. Außerdem läuft auf NFS-Servern der **mountd**-Dämon, der Anfragen zum Einhängen bearbeitet. Auf dem Client wird die Verwaltung des Caches und der Buffer von **biod**, dem Block-I/O-Dämon, vorgenommen. Der **portmap**-Dämon bildet RPC-Programmnummern auf die entsprechenden TCP/IP-Portnummern ab.

Exportieren von Dateisystemen

Um einen NFS-Server einzurichten, sollten Sie zunächst überprüfen, ob der dafür bestimmte Rechner all die Rechner erreichen kann, die Verzeichnisse von ihm importieren wollen. Danach editieren Sie die Datei */etc/exports* und tragen dort all die Dateisysteme mit ihren Einhängepunkten ein, die exportiert werden sollen. Wenn Sie einen Automounter und **mountd** verwenden, werden die Dateien exportiert, falls es die Zugriffsrechte in */etc/exports* erlauben.

Einhängen von Dateisystemen

Um via NFS auf ein entferntes Dateisystem zugreifen zu können, muß es nach dem Start von NFS eingehängt werden, entweder durch Verwenden des **mount**-Befehls oder durch Eintragen des Einhängepunktes und des entfernten Dateisystems in der Datei */etc/fstab*. Der **mount**-Befehl ruft den **mountd**-Dämon des Servers, der die Zugriffsrechte des Clients überprüft und einen Zeiger auf das Dateisystem zurückgibt. Wenn ein Verzeichnis einmal eingehängt ist, bleibt es mit dem lokalen Verzeichnisbaum verbunden, bis es mit dem **umount**-Befehl wieder ausgehängt wird oder der lokale Rechner heruntergefahren wird.

Normalerweise kann nur der Superuser Dateisysteme via NFS einhängen. Man kann jedoch auch normalen Benutzern das Ein- und Aushängen von bestimmten Dateisystemen mit den **mount**- und **umount**-Befehlen erlauben, indem man die Option „user" in der Datei */etc/fstab* einträgt. Damit kann die Netzwerkbelastung reduziert werden, indem Verzeichnisse nur bei Bedarf eingehängt werden. Um das Einhängen durch normale Benutzer einzuschalten, sollten Sie für jedes einzuhängende Dateisystem einen Eintrag in */etc/fstab* anlegen.

Überblick über NIS

NIS bezeichnet einen Dienst, der früher als Sun Yellow Pages (YP) bekannt war. Er wird verwendet, um Konfigurationsinformationen auf allen Rechnern eines Netzwerks einheitlich und konsistent zu halten. Dies wird dadurch erreicht, daß ein einzelner Rechner sämtliche Dateien und Datenbanken zur Systemverwaltung hält und diese Information an alle anderen Rechner des Netzwerkes weitergibt. Die Information wird in Datenbanken übersetzt, den sogenannten Tabellen *(maps)*. NIS basiert auf dem RPC-Protokoll. Zur Zeit sind zwei NIS-Server für Linux frei verfügbar: **yps** und **ypserv**.

Server

Bei NIS gibt es zwei verschiedene Arten von Servern: Master und Slaves. Die Master-Server verwalten die Tabellen und verteilen sie an die Slave-Server, von denen sie von den Clients gelesen werden.

Domänen

Eine NIS-Domäne ist eine Gruppe von Rechnern, die dieselbe Menge an Tabellen benutzt. Die Tabellen sind in einem Unterverzeichnis von */var/yp*, das den gleichen Namen wie die Domäne hat, abgelegt. Die Rechner einer Domäne teilen sich die Informationen über Benutzer, Benutzergruppen und Rechner. NIS-Domänennamen werden mit dem **domainname**-Befehl eingestellt.

NIS-Tabellen

NIS speichert Informationen in Datenbankdateien, den sogenannten *maps* (Tabellen). Jede Tabelle besteht aus einem Paar von **dbm**-Datenbankdateien, von denen die eine ein Verzeichnis der Schlüssel und die andere die eigentlichen Daten enthält. Da **dbm**-Dateien keine Textdateien sind, müssen spezielle NIS-Werkzeuge wie **yppush** verwendet werden, um Tabellen zwischen Rechnern auszutauschen.

Die Datei */var/yp/YP_MAX_X_LATE* enthält eine komplette Aufstellung aller aktiven NIS-Tabellen sowie der NIS-Aliasnamen für NIS-Tabellen. Alle Tabellen müssen hier aufgeführt sein, damit NIS sie verteilen kann.

Werkzeuge zur Tabellenmanipulation

NIS-Tabellen werden mit den folgenden Hilfsprogrammen verwaltet:

makedbm
: Erzeugen von **dbm**-Dateien. Verändert werden sollten nur die *ypservers*-Tabelle und andere nicht-standardisierte Tabellen.

ypinit
: Erzeugen und installieren von NIS-Datenbanken. Bearbeitet die Tabellen, wenn NIS initialisiert wird. Sollte nicht verwendet werden, wenn NIS bereits in Betrieb ist.

yppush
: Transferiert aktualisierte Tabellen vom Master-Server.

Verwalten von NIS

NIS wird durch das Einrichten von NIS-Servern und NIS-Clients eingeschaltet. Die Beschreibungen hier gehen von der Verwendung von **ypserv** aus, was keine Master-/Slave-Server-Konfiguration erlaubt. Alle NIS-Befehle benötigen das RPC **portmap**-Programm. Stellen Sie also sicher, daß es installiert ist und läuft, bevor Sie NIS einrichten.

Einrichten eines NIS-Servers

Zum Einrichten eines NIS-Servers gehören die folgenden Schritte:

1. Einstellen des NIS-Domänennamens mit **domainname**.
2. Editieren Sie die Datei *ypMakefile*, die angibt, welche Datenbanken aus welchen Quellen gebaut werden sollen.
3. Kopieren Sie *ypMakefile* nach */var/yp/Makefile*.
4. Starten Sie **make** im Verzeichnis */var/yp*, wodurch die Datenbanken erzeugt werden und der Server initialisiert wird.
5. Starten von **ypserv**, dem NIS-Server-Dämon.

Einrichten eines NIS-Clients

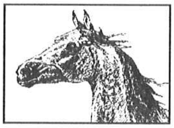
Zum Einrichten eines NIS-Clients gehört: Einstellen des Domänennamens mit **domainname** (der Name muß identisch mit dem beim Server verwendeten sein) und Starten von **ypbind**.

NIS-Benutzerzugänge

In NIS-Netzwerken gibt es zwei Arten von Benutzerzugängen: verteilte und lokale. Verteilte Zugänge müssen vom Master-Rechner verwaltet werden; sie enthalten Informationen, die auf allen Rechnern der NIS-Domäne gleich sind. Änderungen an verteilten Zugängen werden über NIS-Tabellen verteilt. Lokale Zugänge werden vom lokalen Computer verwaltet; sie stellen Informationen zur Verfügung, die spezifisch für den jeweiligen Rechner sind. Diese lokalen Informationen werden nicht von NIS-Tabellen beeinflußt, und Änderungen an ihnen beeinflussen auch NIS nicht. Wenn NIS installiert wird, werden für bereits existierende Zugänge die lokalen Informationen verwendet, falls nichts anderes angegeben wird.

RPC und XDR

RPC (Remote Procedure Call) ist ein Protokoll, das sowohl von NFS als auch von NIS verwendet wird. Es erlaubt einem Rechner, eine Prozedur auf einem entfernten Rechner aufzurufen, als ob sie lokal abgearbeitet werden würde. RPC besteht aus einer Implementation in Form einer Bibliothek von Funktionen und einem Netzwerkstandard für die Datenrepräsentation namens XDR (eXternal Data Representation).



KAPITEL 14

Befehle zur System- und Netzwerkverwaltung

In diesem Kapitel werden die einzelnen Linux-Befehle zur Systemverwaltung beschrieben. Kapitel 13 enthält eine Übersicht, wie diese Befehle mit den Protokollen und Schichten der Netzwerksoftware zusammenspielen. Verwandte Themen finden Sie auch in anderen O'Reilly-Büchern. In *UNIX System Administration* von Æleen Frisch werden die Befehle zur Verwaltung des lokalen Rechners detailliert erläutert, während *TCP/IP Netzwerk Administration* von Craig Hunt und *Linux — Wegweiser für Netzwerker* von Olaf Kirch die TCP/IP-Befehle erläutern. Außerdem behandelt *NFS und NIS — Verwaltung von UNIX-Netzwerken* von Hal Stern NFS- und NIS-Befehle.

Alphabetische Zusammenfassung der Befehle

adduser

adduser *Benutzer*

Anlegen eines *Benutzers*: Definieren einer neuen Benutzer-ID und Eintragen der relevanten Informationen in */etc/passwd*. Sehen Sie in */etc/adduser.conf* nach, um passende Benutzer- und Gruppen-IDs herauszufinden. Weiter wird ein Heimatverzeichnis angelegt, und Konfigurationsdateien werden aus */etc/skel* dorthinein kopiert. Es werden ein Paßwort und Finger-Informationen abgefragt. **adduser** kann nur vom Superuser ausgeführt werden.

agetty

agetty [*Optionen*] *Port Baudrate* [*Terminal*]

Die Linux-Version von **getty**. Stellt Terminaltyp, Übertragungsmodi, Geschwindigkeit und Verbindungsart ein. **agetty** wird von **init** aufgerufen und ist der zweite Prozeß in der Folge **init-getty-login-shell**, die letztlich zum Anmelden eines Benutzers am Linux-System führt. **agetty** liest den Benutzernamen ein und ruft den **login**-Befehl mit dem Benutzernamen als Argument auf. Während der Benutzername eingelesen wird, versucht sich **agetty** an das verwendete System

→

agetty
←

hinsichtlich Übertragungsgeschwindigkeit und des verwendeten Gerätes anzupassen.

Sie müssen einen Port angeben, den **agetty** im */dev*-Verzeichnis sucht. Hier ist auch - möglich; in diesem Fall liest **agetty** von der Standard-Eingabe. Weiterhin müssen Sie eine *Baudrate* angeben. Dies kann eine durch Kommata getrennte Liste von Geschwindigkeiten sein, die von **agetty** der Reihe nach ausprobiert werden. Optional kann ein *Terminal* angegeben werden; dieser Wert überschreibt die TERM-Umgebungsvariable.

Optionen

-h
: Verwenden der Hardware-Flußkontrolle anstelle der softwaremäßigen.

-i
: Unterdrücken der Ausgabe von */etc/issue* vor Ausgabe der Anmeldeaufforderung.

-l *Programm*
: Angeben eines Programms, das anstelle von */bin/login* verwendet wird.

-m
: Versucht, die passende Übertragungsgeschwindigkeit (Baudrate) zu raten.

-t *Timeout*
: Gibt an, daß sich **agetty** beenden soll, wenn das Aufbauen der Verbindung erfolgreich ist, aber danach *Timeout* Sekunden lang keine Eingabe erfolgt.

-L
: Erwartet kein Carrier-Detect-Signal (lokale Verbindung). Diese Option wird verwendet, wenn Terminals direkt angeschlossen sind.

arp

arp [*Optionen*]

TCP/IP-Befehl. Löscht, ergänzt oder gibt den ARP-Cache (*/proc/net/arp*) des Kernels aus.

Optionen

-v
: „Verbose-Modus"

-t *Typ*
: Sucht nach Einträgen dieses Typs im ARP-Cache. *Typ* muß entweder **ether** (IEEE 802.3 10Mbps Ethernet) oder **ax25** (AX.25 Packet Radio) sein; der Default ist **ether**.

-a [*Rechner*]
: Gibt die Einträge für *Rechner* oder alle Einträge — falls kein Rechner angegeben ist — aus.

-d *Rechner*
: Entfernt den Eintrag für *Rechner*.

-s *Rechner Hardware-Adresse*
: Fügt den Eintrag *Rechner Hardware-Adresse* hinzu. **ether**-Adressen sind sechs hexadezimale Bytes, die durch Doppelpunkte voneinander getrennt werden.

-f *Datei*
: Liest Einträge aus *Datei* und fügt sie hinzu.

arp

badblocks [*Optionen*] *Gerät Block–Anzahl*

badblocks

Durchsucht *Gerät* nach schlechten Blöcken, wobei die Anzahl der Blöcke auf dem Gerät angegeben werden muß.

Optionen

-b *Blockgröße*
: Gibt die erwartete Blockgröße an.

-o *Datei* Umlenken der Ausgabe in *Datei.*

-v „Verbose-Modus".

-w Überprüft jeden Block, indem zunächst in ihn hinein geschrieben und danach wieder aus ihm gelesen wird.

bootpd [*Optionen*] [*Konfigurationsdatei* [*Dumpdatei*]]

bootpd

TCP/IP-Befehl. Der Internet Boot Protocol Server. **bootpd** wird normalerweise von */etc/inetd* gestartet, indem die folgende Zeile in die Datei */etc/inetd.conf* aufgenommen wird:

```
bootps dgram udp wait root /etc/bootpd bootpd
```

Damit wird **bootpd** nur gestartet, wenn eine Anfrage an den Server eintrifft. Der Server kann aber auch von der Kommandozeile gestartet werden. Nach dem Starten liest **bootpd** zunächst seine Konfigurationsdatei */etc/bootptab* (oder die auf der Kommandozeile angegebene *Konfigurationsdatei*) und wartet dann auf BOOTREQUEST-Pakete.

bootpd konsultiert */etc/services*, um die Portnummern herauszufinden, die verwendet werden sollen. Dabei werden zwei Einträge gelesen: **bootps** — der Port, auf dem der Server auf Anfragen warten soll — und **bootpc** — der Port, auf dem der Server den Clients antwortet.

Wenn **bootpd** mit der Option –DDEBUG kompiliert worden ist, schreibt er bei Empfang des Signals SIGUSR1 seine Datenbank im Speicher in die Datei */etc/bootpd.dump* oder in die auf der Kommandozeile angegebene *Dumpdatei.*

Optionen

-c *Verzeichnis*
: **bootpd** arbeitet im angegebenen Verzeichnis.

-d *Stufe* Angeben der Debugging-Stufe. Wenn *Stufe* nicht angegeben wird, wird die aktuelle Stufe um eins erhöht.

-t *Timeout*
: Gibt einen Timeout-Wert in Minuten an. 0 bedeutet kein Timeout.

→

bootpd
←

Die Konfigurationsdatei

Im Format der Konfigurationsdatei von **bootpd** beschreiben zweibuchstabige Symbole, bei denen die Groß-/Kleinschreibung relevant ist, verschiedene Parameter der einzelnen Rechner. Diese Parameter werden durch Doppelpunkte getrennt. Das allgemeine Format ist:

```
Rechnername:tg=Wert:tg=Wert:tg=Wert
```

wobei *Rechnername* der Name eines bootp-Clients und *tg* ein Symbol ist, das für ein Tag steht. Die folgenden Tags sind bekannt:

Tags

bf	Bootdatei
bs	Größe der Bootdatei in Blöcken von 512 Oktetten
cs	Adreßliste der Cookie-Server
ds	Adreßliste der DNS-Server
gw	Adreßliste der Gateways
ha	Hardware-Adresse des Rechners
hd	Heimatverzeichnis der Bootdatei
hn	Schicken des Rechnernamens
ht	Hardwaretyp des Rechners (siehe den RFC "Assigned Numbers")
im	Adreßliste der Impress-Server
ip	IP-Adresse des Rechners
lg	Adreßliste der Log-Server
lp	Adreßliste der **lpr**-Server
ns	Adreßliste der IEN-116 Namen-Server
rl	Adreßliste der Server für das Resource Location Protocol
sm	Subnetzmaske des Rechners
tc	Fortsetzungstabelle
to	Zeitverschiebung von UTC in Sekunden
ts	Adreßliste der Time Server
vm	Auswahl des Magic Cookies für den Hersteller

Außerdem gibt es das generische Tag **T***n*, wobei *n* eine Tagnummer des Herstellerfeldes aus dem RFC 1048 ist. Generische Daten können entweder als Folge von Hexadezimalzahlen oder als gequoteter ASCII-String geschrieben werden.

bootpgw

bootpgw [*Optionen*] *Server*

Internet Boot Protocol Gateway. Verwalten eines Gateways, der **bootpd**-Anfragen an den Rechner *Server* weiterleitet. Außer BOOTREPLY-Paketen werden auch BOOTREQUEST-Pakete verarbeitet. **bootpg** wird normalerweise von */etc/inetd* gestartet, indem die folgende Zeile in die Datei */etc/inetd.conf* aufgenommen wird:

```
bootps dgram udp wait root /etc/bootpgw bootpgw
```

Damit wird **bootpgw** nur gestartet, wenn eine entsprechende Anfrage eintrifft. **bootpgw** kennt die gleichen Optionen wie **bootpd** mit Ausnahme von **-c**.

bootptest [*Optionen*] *Server* [*Muster*]

bootptest

TCP/IP-Befehl. Testet den **bootpd**-Dämon von *Server*, indem zehn Sekunden lang, oder bis der Server antwortet, jede Sekunde eine Anfrage geschickt wird. Die Optionen werden aus der *Muster*-Datei gelesen, falls diese angegeben wurde.

Optionen

-f *Datei* Liest den Boot-Dateinamen aus *Datei*.

-h Identifizieren des Clients über die Hardware-Adresse anstelle der IP-Adresse.

-m *Magic-Number*
Stellt *Magic-Number* als erstes Wort des Herstelleroptionen-Feldes zur Verfügung.

cfdisk [*Optionen*] [*Gerät*]

cfdisk

Partitionieren einer Festplatte. Erlaubte Geräte sind */dev/hda* (Default), */dev/hdb*, */dev/sda*, */dev/sdb*, */dev/sdc* und */dev/sdd*. Siehe auch **fdisk**.

Optionen

-a Die aktuelle Partition wird mit dem Cursor anstelle von inverser Darstellung markiert.

-c *Zylinder*
Angeben der Zylinderanzahl.

-h *Köpfe* Angeben der Kopfanzahl.

-s *Sektoren*
Angeben der Sektorenanzahl.

-z Partitionstabelle nicht einlesen. Das Partitionieren wird mit einer leeren Tabelle begonnen.

-P *Format*
Anzeigen der Partitionstabelle im angegebenen Format. Erlaubte Werte sind **r** (rohe Daten), **s** (Sektorenreihenfolge) und **t** (rohes Format).

Befehle

Pfeil nach oben, Pfeil nach unten
Bewegen zwischen den Partitionen.

b Einstellen, ob eine Partition bootbar sein soll oder nicht.

d Löschen der Partition (andere Partitionen können den Platz einnehmen).

g Verändern der Plattengeometrie. Es wird angefragt, was geändert werden soll: Zylinder (c), Köpfe (h) oder Sektoren (s).

h Hilfe.

m Versuchen, die maximale Ausnutzung des vorhandenen Platzes in der Partition sicherzustellen.

→

cfdisk ←

n	Anlegen einer neuen Partition. Es werden weitere Informationen abgefragt.
p	Anzeigen der Partitionstabelle.
q	Beenden, ohne die Änderungen zu speichern.
t	Abfragen eines neuen Dateisystemtyps und Ändern der aktuellen Partition auf diesen Typ.
u	Ändern der Größeneinheit der Partition, wobei zwischen Megabytes, Sektoren und Zylindern hin und her geschaltet wird.
W	Speichern der Änderungen. Das W muß als Großbuchstabe eingegeben werden!

chroot

chroot *NeueWurzel* [*Befehl*]

Ändern des Wurzelverzeichnisses für *Befehl* oder, falls keiner angegeben ist, für eine neue Kopie der Shell des Benutzers. Der Befehl oder die Shell werden relativ zum neuen Wurzelverzeichnis ausgeführt. Die Bedeutung des einleitenden / in Pfadnamen wird für diesen Befehl und alle seine Kinder zu *NeueWurzel* geändert. Außerdem wird das aktuelle Arbeitsverzeichnis auf *NeueWurzel* gesetzt. Dieser Befehl kann nur vom Superuser ausgeführt werden.

clock

clock [*Optionen*]

Einstellen oder Auslesen der CMOS-Uhr.

Optionen

Es muß genau eine der Optionen -a, -r oder -w angegeben werden:

-a	Setzen der Systemzeit auf die CMOS-Zeit.
-r	Ausgeben der aktuellen CMOS-Zeit.
-u	Beim Auslesen oder Einstellen der Uhr wird Universalzeit verwendet. Es muß eine weitere Option angegeben werden.
-w	Setzen der CMOS-Zeit auf die Systemzeit.

Dateien

/etc/adjtime

Enthält Informationen über das Setzen der CMOS-Uhr: die Korrektur in Sekunden pro Tag, die letzte Zeit, zu der die Uhr benutzt wurde, und der Teil einer Sekunde, der nach der letzten Anpassung übrigblieb.

cpio

cpio *Schalter* [*Optionen*]

Kopieren von Archiven von oder auf Festplatte, Diskette oder Band. Bei jedem der drei Schalter -i, -o und -p sind andere Optionen erlaubt.

Funktionsoptionen

-i [*Optionen*] [*Muster*]
: Herausholen von Dateien, die auf die *Muster* passen. Jedes Muster kann Metazeichen für Dateinamen der Bourne-Shell enthalten. (Muster sollten gequotet werden, damit sie von **cpio** und nicht von der Shell interpretiert werden.) Wenn kein Muster angegeben wird, werden alle Dateien extrahiert. Während des Herausholens werden bereits existierende Dateien nicht überschrieben, wenn die Versionen im Archiv älter sind (es sei denn, es wird **-u** angegeben).

-o [*Optionen*]
: Kopieren von Dateien, deren Namen von der Standardeingabe gelesen werden, in das Archiv.

-p [*Optionen*] *Verzeichnis*
: Kopieren von Dateien in ein anderes Verzeichnis auf demselben System. Die Zielpfadnamen werden relativ zu *Verzeichnis* interpretiert.

Übersicht über die gültigen Optionen

Die folgende Tabelle zeigt die gültigen Optionen für die **-i**-, **-o**- und **-p**-Schalter. Der - wird aus Gründen der Übersichtlichkeit jeweils weggelassen.

```
i:   bcdf mnrtsuv B SVCEHMR IF
o: 0a c          vABL VC HM O F
p: 0a  d 1m     uv  L V    R
```

Optionen

-0, --null
: Erwartet, daß die Liste der Dateinamen mit Null anstatt mit einem Zeilenwechsel terminiert ist. Damit können auch Dateien angegeben werden, deren Name einen Zeilenwechsel enthält.

-a, --reset-access-time
: Setzt die Zugriffszeit der Eingabedateien auf den aktuellen Zeitpunkt.

-A, --append
: Anhängen der Dateien an ein existierendes Archiv, bei dem es sich um eine Datei auf einer Festplatte oder Diskette handeln muß. Die Datei wird mit **-O** oder **-F** angegeben.

-b, --swap
: Bytes und Halbwörter vertauschen.

-B Blocken der Eingabe auf 5120 Byte pro Datensatz (der Default ist 512 Byte pro Datensatz).

-c Lesen oder Schreiben der Header-Information als ASCII-Zeichen. Dies ist nützlich, wenn die Quell- und Zielmaschinen von unterschiedlicher Bauart sind.

-C *n*, **--io-size=***n*
: Wie **B**, die Blockgröße kann aber jede positive Ganzzahl *n* sein.

→

SysAdmin Befehle

-d, --make-directories
Fehlende Verzeichnisse werden angelegt.

-E *Datei*, **--pattern-file=***Datei*
Extrahieren der in *Datei* angegebenen Dateinamen.

-f, --nonmatching
Es werden alle Dateien *außer* denen, auf die die Muster passen, kopiert.

-F, --file=*Datei*
Als Archiv wird *Datei* anstelle von **stdin** oder **stdout** verwendet. *Datei* darf auch auf einer anderen Maschine liegen, wenn sie in der Form *Benutzer@Rechnername:Datei* angegeben wird.

--force-local
Es wird nicht angenommen, daß die durch **-F**, **-I** oder **-O** angegebene Datei auf dem entfernten Rechner liegt, auch wenn der Dateiname ein @ enthält.

-H *Typ*, **--format=***Typ*
Verwendet das Format *Typ*. Der Default beim Kopieren in Archive ist **bin**. Die folgenden Formate sind gültig (Groß-/Kleinschreibung darf gemischt werden):

bin	Binär
odc	Altes portables Format (POSIX.1)
newc	Neues portables Format (SVR4)
crc	Neues portables Format (SVR4) mit Prüfsumme
tar	tar
ustar	POSIX.1 tar (erkennt auch GNU-tar-Archive)
hpbin	Binärformat von HP-UX (veraltet)
hpodc	Portables Format von HP-UX

-I *Datei*
Lesen aus dem Archiv *Datei*. Die Datei darf auf einem anderen Rechner liegen (siehe **-F**).

-k Ignoriert. Nur zur Abwärtskompatibilität.

-l, --link
Dateien werden gelinkt anstatt kopiert.

-L, --dereference
Verfolgen symbolischer Links.

-m, --preserve-modification-time
Erhalten der vorherigen Modifikationszeit der Dateien.

-M *Nachricht*, **--message=***Nachricht*
Ausgeben der *Nachricht*, wenn das Medium gewechselt werden muß. Die Zeichenkette **%d** kann für die numerische ID des Mediums verwendet werden. **-M** ist nur möglich, wenn auch **-I** oder **-O** angegeben wurde.

-n, --numeric-uid-gid
Anzeigen der Benutzer- und Gruppen-ID bei der ausführlichen Ausgabe des Inhalts.

--no-preserve-owner
Alle kopierten Dateien gehören demjenigen, der **cpio** aufgerufen hat, nicht dem ursprünglichen Benutzer. Macht nur Sinn, wenn der Befehl vom Superuser aufgerufen wird.

-O *Datei*
Umleiten der Ausgabe nach *Datei*. *Datei* darf auf einer anderen Maschine liegen (siehe -F).

-r Interaktives Umbenenennen der Dateien.

-R [*Benutzer*][:.][*Gruppe*], --owner [*Benutzer*][:.][*Gruppe*]
Ändern der Besitzer- und Gruppeninformation der Datei (kann nur vom Superuser verwendet werden).

-s, --swap-bytes
Vertauschen von Bytes.

-S, --swap-half-words
Vertauschen von Halbwörtern.

-t, --list
Ausgeben eines Inhaltsverzeichnisses der Eingabe (es werden keine Dateien angelegt). Wenn auch -v angegeben wird, ähnelt die Ausgabe **ls -l**.

-u, --unconditional
Unbedingtes Kopieren, ältere Dateien können neuere überschreiben.

-v Ausgeben einer Liste der Dateinamen.

-V, --dot
Ausgeben eines Punktes für jede gelesene oder geschriebene Datei (damit kann der Fortschritt von **cpio** angezeigt werden, ohne den Bildschirm zu sehr durcheinanderzubringen).

Beispiele

Anlegen einer Liste alter Dateien mit **find**; diese Liste wird als Eingabe für **cpio** verwendet:

```
find . -name "*.alt" -print | cpio -ocBv\
    > /dev/rst0
```

Restaurieren aller Dateien vom Band, deren Name **save** enthält (Unterverzeichnisse werden angelegt, falls nötig):

```
cpio -icdv "save" < /dev/rst0
```

Verschieben eines Verzeichnisbaums:

```
find . -depth -print | cpio -padm /neuerort
```

cron

cron

Wird normalerweise während des Systemstarts aufgerufen. cron führt bestimmte Befehle, die in der *crontab* des jeweiligen Benutzers in

→

SysAdmin Befehle

cron ←	*/var/cron/tabs* stehen, zu vorgegebenen Zeiten aus. Die *crontab*-Dateien tragen dabei den Namen des Benutzers und werden mit Hilfe des Befehls **crontab** bearbeitet.

debugfs

debugfs [[*Option*] *Gerät*]

Debuggen eines **ext2**-Dateisystems. *Gerät* ist die Gerätedatei, die zu dem Gerät gehört, welches das **ext2**-Dateisystem enthält (z.B. */dev/hda3*).

Option

-w
: Schreiben auf das Dateisystem ermöglichen.

Befehle

cat *Datei*
: Ausgeben des Inhalts einer Inode auf die Standardausgabe.

cd *Verzeichnis*
: Wechseln des aktuellen Arbeitsverzeichnisses.

chroot *Verzeichnis*
: Das Wurzelverzeichnis auf die angegebene Inode setzen.

close
: Schließen des gerade geöffneten Dateisystems.

clri *Datei*
: Löschen des Inhalts der Inode, die zu *Datei* gehört.

dump *Datei Ausgabedatei*
: Ausgeben des Inhalts einer Inode in die *Ausgabedatei.*

expand_dir *Verzeichnis*
: Expandieren des Verzeichnisses.

find_free_block [*Ziel*]
: Finden und Allozieren des ersten freien Blocks nach *Ziel.*

find_free_inode [*Verz* [*Modus*]]
: Finden und Allozieren einer freien Inode.

freeb *Block*
: Markiert *Block* als nicht alloziert.

freei *Datei*
: Gibt die zu *Datei* gehörende Inode frei.

help
: Ausgeben einer Liste der Befehle, die von **debugfs** verstanden werden.

icheck *Block*
: Durchführen einer Block-Inode-Übersetzung.

initialize *Gerät Blockgröße*
: Anlegen eines **ext2**-Dateisystems auf dem angegebenen *Gerät.*

kill_file *Datei*
: Entfernen der angegebenen *Datei* und Freigeben aller ihrer Blöcke.

ln *Quelldatei Zieldatei*
: Anlegen eines Links.

ls [*Pfadname*]
Emulieren des **ls**-Befehls.

modify_inode *Datei*
Modifizieren des Inhalts der zu *Datei* gehörenden Inode.

mkdir *Verzeichnis*
Anlegen des *Verzeichnisses.*

mknod *Datei* [**p**|[[**c**|**b**] *Majornummer Minornummer*]]
Anlegen einer Gerätedatei.

ncheck *Inode*
Durchführen einer Inode-Namen-Übersetzung.

open [**-w**] *Gerät*
Öffnen eines Dateisystems.

pwd Aktuelles Arbeitsverzeichnis ausgeben.

quit **debugfs** beenden.

rm *Datei* *Datei* entfernen.

rmdir *Verzeichnis*
Verzeichnis entfernen.

setb *Block* *Block* als belegt markieren.

seti *Datei* Die zu *Datei* gehörende Inode als in Benutzung markieren.

show_super_stats
Inhalt des Superblocks ausgeben.

stat *Datei* Inhalt der zu *Datei* gehörenden Inode ausgeben.

testb *Block* Überprüfen, ob *Block* als alloziert markiert ist.

testi *Datei* Überprüfen, ob die zu *Datei* gehörende Inode als alloziert markiert ist.

unlink *Datei*
Entfernen eines Links.

write *Quelldatei Datei*
Anlegen von *Datei* im Dateisystem. Der Inhalt von *Quelldatei* wird in die neu angelegte Datei kopiert.

debugfs

depmod [*Optionen*] *Module*

depmod

Anlegen einer Datei mit Abhängigkeiten für die auf der Kommandozeile angegebenen Module. Diese Datei kann von **modprobe** verwendet werden, um automatisch alle nötigen Module nachzuladen. Normalerweise wird die Zeile **/sbin/depmod -a** in einer der Dateien in */etc/rc.d* aufgenommen, damit die Modulabhängigkeiten nach dem Hochfahren des Systems zur Verfügung stehen.

Optionen

-a Anlegen der Abhängigkeiten für alle in */etc/conf.modules* eingetragenen Module.

-d Debug-Modus. Alle ausgeführten Befehle werden angezeigt.

→

depmod ←	-e Nicht aufgelöste Symbole anzeigen. -v Verarbeitete Module anzeigen. *Dateien* */etc/conf.modules* Informationen über Module: Welche Module andere benötigen und welche Verzeichnisse zu welchem Typ Modul gehören. */sbin/insmod, /sbin/rmmod* Diese Programme werden von **depmod** verwendet.
dmesg	**dmesg** [*Optionen*] Anzeigen der Kontrollmeldungen des Systems aus dem Ringbuffer des Kernels. In diesem Buffer sind alle Kernelmeldungen seit dem letzten Hochfahren oder, falls der Buffer voll war, nur die neuesten abgelegt. *Optionen* -c Löschen des Buffers nach der Ausgabe der Meldungen. -n *Stufe* Gibt an, von welcher Stufe die Systemmeldungen ausgegeben werden sollen.
dnsdomainname	**dnsdomainname** TCP/IP-Befehl. Ausgeben des DNS-Domänennamen des Systems. Siehe auch **hostname**.
domainname	**domainname** [*Name*] NFS/NIS-Befehl. Setzen oder Anzeigen der aktuellen NIS-Domäne. Wenn kein Argument angegeben wird, zeigt **domainname** die aktuelle NIS-Domäne an. Das Einstellen der Domäne durch Angabe eines Argumentes ist dem Superuser vorbehalten.
dumpe2fs	**dumpe2fs** *Gerät* Ausgeben von Informationen über den Superblock und die Blockgruppen von *Gerät*.
e2fsck	**e2fsck** [*Optionen*] *Gerät* **fsck.ext2** [*Optionen*] *Gerät* Ähnlich wie **fsck**, aber speziell für das Second Extended Dateisystem von Linux. (fsck ruft fsck.ext2 auf, wenn es sich um ein Second Extended handelt.) *Optionen* -b *Superblock* Verwenden von *Superblock* anstelle des Default-Superblocks.

-d Debugging-Modus.

-f Erzwingt ein Überprüfen auch dann, wenn der Kernel das Dateisystem bereits als gültig markiert hat. Normalerweise beendet sich **e2fsck** ohne Überprüfungen, wenn der Kernel das Dateisystem als sauber akzeptiert hat.

-l *Datei*
: Aus *Datei* wird eine Liste mit defekten Blöcken ausgelesen. Zusätzlich wird auch das Gerät nach defekten Blöcken abgesucht.

-n Sicherstellen, daß keine Änderungen am Dateisystem vorgenommen werden. Jede Rückfrage des Programms wird automatisch mit „nein" beantwortet.

-p („Putzen") Nicht-interaktives Reparieren aller defekten Blöcke.

-t Ausgeben von Zeit-Statistiken.

-v Verbose-Modus.

-y Automatisches Beantworten aller Rückfragen mit „ja".

-B *Größe*
: Erwartet, daß der Superblock die angegebene *Größe* hat. Wenn dies nicht der Fall ist, beendet sich **e2fsck**.

-F Leeren der Buffer-Caches vor der Überprüfung.

-L *Datei*
: Aus *Datei* wird eine Liste defekter Blöcke entnommen, das Gerät wird nicht darüber hinaus nach weiteren defekten Blöcken abgesucht.

fdisk [*Optionen*] [*Gerät*]

Verwalten von Festplattenpartitionen über ein Menü. **fdisk** gibt Informationen über die Partitionen aus, legt Partitionen an, löscht sie wieder und legt die aktive Partition fest. Auf jede der vier (physikalischen) Partitionen kann ein anderes Betriebssystem installiert werden, es kann aber nur jeweils eine Partition aktiv sein. Die kleinste empfohlene Größe für eine Linux-Systempartition beträgt 40 Megabyte. *Gerät* ist normalerweise */dev/hda*, */dev/hdb*, */dev/sda*, */dev/sdb*, */dev/hdc*, */dev/hdd*, etc. Siehe auch **cfdisk**.

Optionen

-l Partitionstabellen ausgeben und Programm beenden.

-s*Partition*
: Größe der *Partition* ausgeben, wenn es sich nicht um eine DOS-Partition handelt.

Befehle

a Aktuelle Partition bootbar oder nicht-bootbar schalten.

d Aktuelle Partition löschen.

l Alle bekannten Partitionstypen aufführen.

m Hauptmenü.

→

fdisk
←

- n — Anlegen einer neuen Partition, es wird nach weiteren Informationen gefragt.
- p — Ausgeben einer Liste der Partitionen einschließlich Informationen über jede.
- q — Beenden; nicht speichern.
- t — Ändern des Partitionstyps der aktuellen Partition.
- u — Ändern der bei Anzeigen und Eingaben verwendeten Einheit. Mögliche Werte sind Zylinder und Sektoren.
- v — Überprüfen: Die Partitionstabelle wird auf Fehler überprüft und die Summe des nicht zugewiesenen Speicherplatzes ausgegeben.
- w — Änderungen speichern und Programm beenden.

---

fingerd

in.fingerd [*Option*]

TCP/IP-Befehl. Server für Informationen über Benutzer dieses Rechners. **fingerd** stellt eine Netzwerkschnittstelle zum **finger**-Programm dar. **fingerd** wartet am finger-Port auf TCP-Verbindungen, liest bei jeder Verbindung genau eine Eingabezeile, reicht die Zeile an **finger** weiter und schickt die Ausgabe von **finger** wieder an den Aufrufenden zurück. **fingerd** wird von **inetd** gestartet und muß daher in der Konfigurationsdatei von **inetd**, */etc/inetd.conf*, eingetragen sein.

Option

- -w — Ausgeben zusätzlicher Informationen wie Systemlaufzeit und Name des Betriebssystems.

---

fsck

fsck [*Optionen*] [*Dateisystem*] ...

Ruft das Überprüfungs- und Reparaturprogramm des jeweiligen Dateisystemtyps auf. Wenn ein Dateisystem in einem ordnungsgemäßen Zustand ist, werden die Anzahl der Dateien und die Anzahl der belegten Blöcke und der freien Blöcke ausgegeben. Wenn ein Dateisystem in einem inkonsistenten Zustand ist, fragt **fsck** vor jedem Reparaturversuch nach. Der Rückgabewert von **fsck** ist die Summe der folgenden möglichen Zustände:

- 1 — Es wurden Fehler gefunden und behoben.
- 2 — Neustarten des Systems empfohlen.
- 4 — Es wurden Fehler gefunden, aber nicht behoben.
- 8 — Ein interner Fehler in **fsck** ist aufgetreten.
- 16 — **fsck** wurde falsch aufgerufen.
- 128 — Ein Fehler im Zusammenhang mit einer dynamischen Bibliothek ist aufgetreten.

Optionen

- -- — Alle folgenden Optionen werden an das dateisystem-spezifische Programm weitergereicht, genau wie alle Optionen, die **fsck** selbst nicht kennt.

fsck

- -r Interaktiver Modus: Vor jedem Reparaturversuch wird zurückgefragt.
- -s Serieller Modus.
- -t *DS-Typ*
  Angeben des Dateisystemtyps. Dateisysteme anderer Typen werden nicht überprüft.
- -A Überprüfen aller in */etc/fstab* aufgeführten Dateisysteme.
- -N Normale Ausführung unterdrücken; es wird lediglich ausgegeben, was gemacht werden würde.
- -R Nur sinnvoll mit -A: Alle Dateisysteme in */etc/fstab* mit Ausnahme des Wurzel-Dateisystems werden überprüft.
- -T Startmeldung nicht ausgeben.
- -V Verbose-Modus.

fsck.minix

fsck.minix [*Optionen*] *Gerät*

Ähnlich wie **fsck**, aber speziell für MINIX-Dateisysteme.

Optionen

- -l Ausgeben der Dateisysteme.
- -r Interaktiver Modus: Vor jedem Reparaturversuch wird zurückgefragt.
- -a Automatischer Modus: Reparieren ohne Rückfragen.
- -v Verbose-Modus.
- -s Informationen über die Superblöcke ausgeben.
- -m MINIX-ähnliche *"mode not cleared"*-Warnungen einschalten.
- -f Erzwingt ein Überprüfen auch dann, wenn der Kernel das Dateisystem bereits als gültig markiert hat. Normalerweise beendet sich **fsck.minix** ohne Überprüfungen, wenn der Kernel das Dateisystem als sauber akzeptiert hat.

ftpd

in.ftpd [*Optionen*]

TCP/IP-Befehl. Internet File Transfer Protocol Server. Der Server verwendet das TCP-Protokoll und wartet auf dem in der ftp-Spezifikation angegebenen Port. **ftpd** wird von **inetd** gestartet und muß einen Eintrag in der Konfigurationsdatei von **inetd**, */etc/inetd.conf*, haben.

Optionen

- -d Debugging-Informationen in das Systemlog schreiben.
- -l Jede FTP-Sitzung im Systemlog mitprotokollieren.
- -T*MaxTimeout*
  Setzen der maximalen Timeout-Periode. Der Default ist zwei Stunden.
- -t*Timeout*
  Setzen der Timeout-Periode auf *Timeout* Sekunden.

SysAdmin Befehle

fuser

fuser [*Optionen*] [*Dateien* | *Dateisysteme*]

Identifizieren der Prozesse, die eine Datei oder ein Dateisystem benutzen. **fuser** gibt die Prozeß-IDs aller Prozesse aus, die *Dateien* oder *Dateisysteme* benutzen. Jeder Prozeß-ID folgt ein einbuchstabiger Code: **c**, wenn der Prozeß die Datei als aktuelles Verzeichnis benutzt; **e**, wenn der Prozeß die Datei ausführt; **f**, wenn der Prozeß die Datei geöffnet hat; **m**, wenn er die Datei als dynamische Bibliothek angezogen hat, und **r**, wenn die Datei das Wurzelverzeichnis des Prozesses ist. Jeder Benutzer, der */dev/mem* und */dev/kmem* lesen darf, kann **fuser** benutzen, aber nur der Superuser kann die Prozesse anderer Benutzer beenden. **fuser** funktioniert nicht mit Dateien, auf die über NFS zugegriffen wird.

Wenn mehr als eine Gruppe von Dateien angegeben wird, können die Optionen für jede Gruppe von Dateien neu angegeben werden. Ein einzelner Querstrich (-) schaltet alle aktuellen Optionen ab; die nächsten Optionen gelten dann für die nächste Gruppe von Dateien.

Optionen

- Alle Optionen auf ihre Defaultwerte zurücksetzen.

-*Signal* Das angegebene *Signal* anstelle von SIGKILL schicken.

-a Informationen über alle angegebenen Dateien ausgeben, selbst wenn kein Prozeß auf sie zugreift.

-k Jedem Prozeß das SIGKILL-Signal schicken.

-l Ausgeben der Signalnamen.

-m Erwartet, daß die *Dateien* auf eingehängten Dateisystemen liegen, und überprüft alle Dateien, die auf das Dateisystem zugreifen.

-s Stilles Arbeiten.

-u Der Prozeß-ID wird der Anmeldename des Benutzers in Klammern nachgestellt.

-v Verbose-Modus.

gated

gated [*Optionen*]

TCP/IP-Befehl. Gateway-Routing-Dämon. **gated** verarbeitet mehrere Routing-Protokolle und ersetzt **routed** und andere Routing-Dämonen, welche die Protokolle Hello, EGP und BGP sprechen. **gated** kann die Routing-Protokolle RIP, BGP, EGP, Hello und OSPF bedienen und auf eines oder mehrere dieser Protokolle eingestellt werden.

Optionen

-c Durchsuchen der Konfigurationsdatei nach syntaktischen Fehlern. Danach beendet sich **gated** und hinterläßt eine Dumpdatei in */usr/tmp/gated_dump*.

-f *Konfigurationsdatei*
Verwenden einer anderen *Konfigurationsdatei*. Der Default ist */etc/gated.conf*.

-n Die Routing-Tabellen des Kernels nicht verändern.

-t [*Trace-Optionen*]
: **gated** mit den angegebenen Tracing-Optionen starten. Wenn keine Schalter angegeben werden, wird **general** angenommen. Die zur Verfügung stehenden Schalter sind:

adv	Policy-Blöcke verwalten
all	Umfaßt **normal**, **policy**, **route**, **state**, **task** und **timer**
general	Umfaßt **normal** und **route**
iflist	Die Schnittstellenliste des Kernels
normal	Normale Protokollinstanzen
parse	Lexikalisches Analyse- und Parseprogramm
policy	Instanzen, in denen eine Policy auf im- oder exportierte Routen angewendet wird
route	Jede Veränderung der Routing-Tabelle
state	Übergänge der Zustandsmaschine
symbols	Aus dem Kernel gelesene Symbole — beachten Sie bitte, daß diese Symbole gelesen werden, bevor die Konfigurationsdatei geparst worden ist, so daß diese Option auf der Kommandozeile angegeben werden muß
task	Systemtasks und -schnittstellen
timer	Timer-Benutzung

-C
: Die Konfigurationsdatei wird auf Fehler überprüft und das Programm beendet. Wenn Fehler auftraten, wird 1 zurückgegeben, ansonsten 0.

-N
: Das Programm nicht automatisch in den Hintergrund legen.

gdc [*Optionen*] *Befehl*

TCP/IP-Befehl. Verwalten von **gated**. Es stehen verschiedene Befehle zum Starten und Beenden des Dämons, zum Schicken von Signalen, zur Verwaltung der Konfigurationsdateien und zum Verwalten von Zuständen und Core-Dumps zur Verfügung.

Optionen

-c *Größe* Angeben der maximalen Core-Dump-Größe.

-f *Größe* Angeben der maximalen Dumpdatei-Größe.

-m *Größe*
: Angeben der maximalen Datensegmentgröße.

-n
: Unterdrücken der Bearbeitung der Kerneltabelle mit Weiterleitungsinformationen.

-q
: Stiller Modus: Warnungen werden unterdrückt, Fehlermeldungen an den Systemlog anstelle von Standardfehler geschickt.

-s *Größe* Angeben der maximalen Stackgröße.

-t *Sekunden*
: *Sekunden* (Default: 10) auf das Vollenden der zum Starten und Anhalten angegebenen Operationen warten.

→

SysAdmin Befehle

Befehle

BACKOUT
: Ersetzt */etc/gated.conf* durch */etc/gated.conf–*, unabhängig davon, ob diese existiert.

backout
: Ersetzt */etc/gated.conf* durch */etc/gated.conf–*, sofern diese existiert.

checkconf
: Gibt Meldungen über syntaktische Fehler in */etc/gated.conf* aus.

checknew
: Gibt Meldungen über syntaktische Fehler in */etc/gated.conf+* aus.

COREDUMP
: Erzwingt einen Core Dump und beendet **gated**.

createconf
: Legt eine leere Datei */etc/gated.conf+* an, sofern diese noch nicht existiert, und setzt diese auf Zugriffsrechte 664, Besitzer **root**, Gruppe **gdmaint**.

dump
: Erzwingt einen Core Dump von **gated** in */usr/tmp/gated_dump* und setzt danach die normale Abarbeitung fort.

interface
: Neuladen der Schnittstellenkonfiguration.

KILL
: Unmittelbar und ohne Sicherung abbrechen.

modeconf
: Setzen aller Konfigurationsdateien auf Modus 664, Besitzer **root**, Gruppe **gdmaint**.

newconf
: Sofern */etc/gated.conf+* existiert, wird diese Datei nach */etc/gated.conf* verschoben. Die alte */etc/gated.conf* wird in */etc/gated.conf–* umbenannt.

reconfig
: Neuladen der Konfigurationsdatei.

restart
: Anhalten und Neustarten von **gated**.

rmcore
: Entfernen aller Core-Dateien von **gated**.

rmdmp
: Entfernen aller Zustandsdumps von **gated**.

rmparse
: Entfernen aller **gated**-Dateien, die Meldungen über syntaktische Fehler enthalten. Diese werden durch die **gdc checkconf**- und **gdc checknew**-Befehle angelegt.

running
: Rückgabe von 0, wenn **gated** läuft, und sonst 1.

start
: **gated** starten, sofern dieser noch nicht läuft. In diesem Fall wird eine Fehlermeldung ausgegeben.

stop
: **gated** so schonend wie möglich anhalten.

term
: Schonend beenden.

toggletrace
: Tracing ein- oder ausschalten.

Dateien

/etc/gated.conf+
: Die Text-Konfigurationsdatei. Wenn Sie mit der Funktion zufrieden sind, können Sie sie mit **gdc newconf** als */etc/gated.conf* installieren.

/etc/gated.conf–
: Sicherungsdatei der alten Konfigurationsdatei.

/etc/gated.conf––
: Sicherungsdatei der Sicherungsdatei der alten Konfigurationsdatei.

/etc/gated.conf
: Eigentliche Konfigurationsdatei.

/etc/gated.pid
: Prozeß-ID von **gated**.

/usr/tmp/gated_dump
: Die Zustandsdump-Datei.

/usr/tmp/gated_parse
: Eine Liste der syntaktischen Fehler, die beim Lesen der Konfigurationsdatei gefunden wurden.

getty [*Optionen*] *Port* [*Geschwindigkeit* [*Terminal* [*Verbindungsart*]]]

Einstellen des Terminaltyps, der Modi, der Übertragungsraten und der Verbindungsart. Auf Linux-Systemen kann auch **agetty** verwendet werden, der eine andere Syntax benutzt. **getty** wird von **init** aufgerufen und ist der zweite Prozeß in der **init-getty-login-shell**-Folge, an deren Ende ein Benutzer sich am Linux-System anmeldet. **getty** liest den Anmeldenamen des Benutzers ein und ruft den **login**-Befehl mit dem Benutzernamen als Parameter auf. Während des Einlesens des Namens versucht **getty**, sich dem System hinsichtlich Geschwindigkeit und des verwendeten Gerätetyps anzupassen.

Sie müssen einen *Port* angeben, anhand dessen sich **getty** mit dem Gerät */dev/Port* verbindet. **getty** liest danach seine Defaultdatei, üblicherweise */etc/default/getty*, mit Laufzeitwerten und -parametern ein. Die meisten von ihnen können auch auf der Kommandozeile angegeben werden, aber die Werte in der Defaultdatei haben Vorrang. Anhand des *Geschwindigkeit*-Arguments wird ein Eintrag in der Datei */etc/gettydefs* ermittelt, der die anfängliche Baudrate, die tty-Einstellungen und die Anmeldeaufforderung, die erreichte Geschwindigkeit und die Verbindungseinstellungen enthält. Der erste Eintrag in */etc/gettydefs* ist der Default. *Terminal* gibt den Terminaltyp an; *Verbindungsart* die (optionale) *line discipline*.

Optionen

-c *Datei*
: Überprüfen einer *gettydefs*-Datei. Die Inhalte und eventuelle Fehlermeldungen werden auf die Standardausgabe ausgegeben.

→

getty ←

- -d *Datei* Verwenden einer anderen Defaultdatei.
- -h Kein Auflegen am Port erzwingen, während **getty** sich initialisiert.
- -r *Pause* Wartet auf ein einzelnes Zeichen am Port und macht danach *Pause* Sekunden Pause.
- -t *Timeout*
  Wenn innerhalb von *Timeout* Sekunden kein Benutzername eingegeben wird, wird die Verbindung geschlossen.
- -w *String*
  Wartet vor der weiteren Abarbeitung auf die Zeichen *String* vom Port.

---

halt

halt [*Optionen*]

In */var/log/wtmp* wird ein Vermerk eingetragen. Wenn der Runlevel des Systems 0 oder 6 ist, werden alle Prozesse angehalten, ansonsten wird **shutdown -nf** aufgerufen.

Optionen

- -d Schreiben nach */var/log/wtmp* unterdrücken.
- -f **halt** auch dann aufrufen, wenn normalerweise **shutdown -nf** aufgerufen werden würde, d.h., ein Aufruf von **halt** wird auch dann erzwungen, wenn sich das System nicht in Runlevel 0 oder 6 befindet.
- -n Den üblichen Aufruf von **sync** nicht ausführen.
- -w Unterdrücken der normalen Operation; es wird lediglich nach */var/log/wtmp* geschrieben.

---

host

host [*Optionen*] *Name* [*Server*]

TCP/IP-Befehl. Ausgeben von Informationen über Rechner und Zonen; Übersetzungen von Rechnernamen und IP-Adressen und umgekehrt (*Name* darf beides sein). Weitere Informationen finden Sie in *DNS und BIND* von Paul Albitz und Cricket Liu.

- -a Wie -t ANY.
- -A Bei Rechnern wird die Authentizität von Namen und IP-Adressen überprüft. Bei Zonen wird für jede aufgeführte Adresse ein umgekehrtes Nachschlagen (Rechnername zur IP-Adresse) durchgeführt.
- -c *Klasse*
  Sucht nur nach Datensätzen des Typs *Klasse*. Mögliche Werte für *Klasse* sind **IN**, **INTERNET**, **CS**, **CSNET**, **CH**, **CHAOS**, **HS**, **HESIOD**, **ANY** und *. Der Default ist **IN**.
- -C Ähnlich wie -l, aber alle Antworten werden daraufhin überprüft, ob sie autoritativ sind. Falls nicht, wird eine entsprechende Meldung ausgegeben.

- **-d** Debugging-Modus.
- **-dd** Extra-Debugging-Modus.
- **-D** Ähnlich wie **-h**, enthält aber auch doppelte Rechner.
- **-e** Unterdrücken von Ausgaben für Rechner, die nicht in der aufgeführten Zone sind, sowie einiger weiterer Datensätze.
- **-E** Ähnlich wie **-h**, enthält aber auch Rechner, die keiner Zone zugeordnet werden können.
- **-f** *Datei*
  Schreiben der Ausgaben nach *Datei* und auf die Standardausgabe.
- **-F** *Datei*
  Schreiben der normalen Ausgabe nach *Datei* und der Ausgaben von Extra-Ressource-Datensätzen auf die Standardausgabe.
- **-G** Ähnlich wie **-H**, enthält aber auch Gateway-Rechner (Rechner mit mehreren Adressen).
- **-H** Interpretiert *Name* als Zone und gibt eine Liste aller eindeutigen Rechnernamen innerhalb dieser Zone aus.
- **-i** Erwartet, daß *Name* eine numerische Adresse ist, und gibt den dazugehörigen Rechnernamen aus.
- **-I** *Zeichen*
  Nimmt an, daß *Zeichen* legale Zeichen im Domänennamen sind (zusätzlich zu den normalen legalen Zeichen).
- **-l** *Zone*
  Erwartet kein *Name*-Argument und gibt die Information für *Zone* aus.
- **-L** *Ebenen*
  Ausgeben rekursiver Zonenlisten für *Ebenen* Zonenebenen.
- **-m** Wie **-t MAILB**.
- **-o** Keine Informationen von Ressource-Datensätzen auf die Standardausgabe ausgeben.
- **-p** Nur den primären Nameserver einer Zone abfragen, falls ein Zonentransfer stattfindet.
- **-P** *Server*
  Nur die in *Server* (eine durch Kommata getrennte Liste) angegebenen Nameserver abfragen, falls ein Zonentransfer stattfindet.
- **-q** Stilles Arbeiten. Alle normalen Warnungen werden unterdrückt.
- **-r** Unterdrücken der Nameserver-Rekursion.
- **-R** Nicht voraussetzen, daß *Name* vollqualifiziert ist. Wenn der Name nicht wie angegeben existiert, wird versucht, den vollen Namen zu ermitteln.
- **-s** *Sekunden*
  Timeout, wenn der Nameserver nicht innerhalb von *Sekunden* Sekunden antwortet.

→

host
←

- -S Ausgeben von Statistiken.
- -t *Typ* Nur Informationen aus Ressource-Datensätzen des Typs *Typ* angeben. Mögliche Werte sind **A**, **NS**, **PTR**, **ANY** und *.
- -T Time-to-live-Werte mit ausgeben.
- -u Virtual Circuits anstelle von Datagrammen bei der Abfrage verwenden.
- -v Verbose-Modus. Ausgeben sämtlicher Informationen aus Ressource-Datensätzen.
- -vv Steigerung des Verbose-Modus. Noch mehr Informationen ausgeben.
- -w Setzen des Timeouts auf unendlich.
- -Z Ausgeben der Datensätze in vollem Zonendateiformat.

icmpinfo

icmpinfo [*Optionen*]

TCP/IP-Befehl. Fängt ICMP-Pakete ab und interpretiert sie. Ausgegeben werden: Adresse und Name des Senders, Ausgangsport, Zielport, Sequenznummer und Paketgröße. Defaultmäßig werden nur Informationen über Pakete, die sich merkwürdig verhalten, ausgegeben.

Optionen

- -k Beenden des **syslogd**-Prozesses, der durch **-l** gestartet wurde.
- -l Protokollieren via **syslogd**. Diese Option kann nur vom Superuser verwendet werden.
- -n IP-Adressen anstelle von Hostnamen verwenden.
- -p Decodieren der Portnummer unterdrücken: Es wird nicht versucht, den Namen des Services, der auf diesem Port lauscht, herauszufinden.
- -s IP-Adresse der Schnittstelle ausgeben, die das Paket empfangen hat, falls es mehrere Schnittstellen auf dem Rechner gibt.
- -v Verbose-Modus. Auch Informationen über normale ICMP-Pakete ausgeben. Noch mehr Informationen können Sie mit -vv und -vvv erhalten.

identd

in.identd [*Optionen*] [*Kerneldatei* [*Kmemdatei*]]

TCP/IP-Befehl. Stellt den Namen des Benutzers, dessen Prozeß die angegebene TCP/IP-Verbindung verwendet, zur Verfügung. Sie können den Kernel und sein Speicherabbild angeben.

Optionen

-a *IP-Adresse*
: Bindet das Socket an die *IP-Adresse*. Nur sinnvoll im Zusammenhang mit -b. Defaultmäßig wird an die Adresse INADDR_ANY gebunden.

- `-b` Von der Kommandozeile aus starten; darf nicht zusammen mit **inetd** verwendet werden.
- `-d` Debugging-Anfragen zulassen.
- `-g`*GID* Versucht, in der Gruppe *GID* zu arbeiten. Nur sinnvoll mit `-b`.
- `-i` Als Dämon starten; pro Anfrage wird ein Prozeß gestartet.
- `-l` Meldungen via **syslogd** ausgeben.
- `-m` Mehrere Anfragen pro Sitzung zulassen.
- `-n` Benutzer-IDs anstelle von Benutzernamen ausgeben.
- `-N` Keinen Benutzernamen und keine Benutzer-ID angeben, wenn im Heimatverzeichnis des Benutzers die Datei *.noident* existiert.
- `-o` Immer OTHER zurückgeben, wenn nach dem Typ des Betriebssystems gefragt wird.
- `-p`*Port* An *Port* lauschen statt am Default-Port 113.
- `-t`*Sekunden*
  Beenden, wenn nach *Sekunden* Sekunden keine neuen Anfragen eingetroffen sind. Beachten Sie, daß in diesem Fall bei Verwendung von `-i` oder `-w` die nächste Anfrage in einem neuen **identd**-Prozeß resultiert. Der Default für *Sekunden* ist unendlich (nie beenden).
- `-u`*UID* Versucht, als Benutzer *UID* zu starten. Nur sinnvoll mit `-b`.
- `-V` Ausgeben der Versionsnummer und Beenden.
- `-w` Als Dämon starten; ein Prozeß bedient alle Anfragen.

imapd

TCP/IP-Befehl. Der Interactive Mail Access Protocol (IMAP) Server Dämon. **imapd** wird von **inetd** gestartet und lauscht auf Port 143 den Anfragen von IMAP-Clients. IMAP ermöglicht es Mailprogrammen, auf entfernte Mailboxen zuzugreifen und sie zu verwalten, als ob sie lokal vorhanden wären. IMAP kann zum Online- und Offline-Lesen von Mail verwendet werden. Das bekannte Mailprogramm Pine enthält IMAP-Unterstützung.

ifconfig [*Schnittstelle*]
ifconfig [*Schnittstelle Adreßfamilie Parameter Adressen*]

TCP/IP-Befehl. Zuweisen einer Adresse an eine Netzwerkschnittstelle (z.B. eine Ethernetkarte oder eine PPP-Verbindung) oder Konfigurieren von Netzwerkschnittstellenparametern. **ifconfig** wird typischerweise beim Hochfahren des Systems benutzt, um die Netzwerkadresse jeder vorhandenen Schnittstelle festzulegen. Außerdem können mit **ifconfig** später die Adresse der Schnittstelle oder andere Betriebsparameter geändert werden. Ohne Argumente gibt **ifconfig** die aktuelle Konfiguration aller aktiven Netzwerkschnittstellen aus. Wird nur eine Netzwerkschnittstelle als einziges Argument angegeben, wird deren Konfiguration ausgegeben.

→

ifconfig
←

Argumente

Schnittstelle

Ein String der Form *Name Einheit*, z.B. *eth0*.

Adreßfamilie

Da eine Schnittstelle Daten in verschiedenen Protokollen empfangen kann, die alle unterschiedliche Adressierungen verwenden, kann mit *Adreßfamilie* die Interpretation der weiteren Parameter eingestellt werden. Erlaubte Werte sind **inet** (TCP/IP), ax25 (AX.25 Packet Radio), **ddp** (Appletalk Phase 2) und **ipx** (Novell). Der Default ist **inet**.

Parameter

Mit **ifconfig** können die folgenden Parameter eingestellt werden:

allmulti/-allmulti

Ein- oder Ausschalten des Sendens eingehender Frames an die Netzwerkschicht des Kernels.

arp/-arp

Ein- oder Ausschalten der Nutzung des Address Resolution Protocols, um Abbildungen zwischen Netzwerk-(Software-)Adressen und Hardware-Adressen herzustellen.

broadcast

(Nur bei **inet**.) Angeben der Adresse, die für Broadcasts im Netzwerk verwendet wird. Der Default ist die Adresse der Schnittstelle, bei der im Rechneranteil alle Bits auf 1 gesetzt sind, also z.B. x.y.z.255 bei einem Klasse-C-Netzwerk.

debug/-debug

Ein- oder Ausschalten des treiber-abhängigen Debugging-Codes.

dest_address

Angeben der Adresse des Gegenübers am anderen Ende einer Point-to-Point-Verbindung.

down

Schaltet eine Schnittstelle vorübergehend ab.

hw *Klasse Adresse*

Einstellen von Klasse und Adresse der Hardware. *Klasse* ist entweder **ether** (Ethernet), **ax25** (AX.25 Packet Radio) oder **ARCnet**.

irq *Adr*

Einstellen der IRQ-Leitung des Geräts.

metric *n*

Einstellen der Routing-Metrik der Schnittstelle auf *n*. Der Default ist 0.

mtu *num*
Einstellen der Maximum Transfer Unit (MTU) der Schnittstelle.

multicast
Setzen des Multicast-Schalters.

netmask *Maske*
(Nur bei **inet**.) Gibt an, wieviel von der Adresse reserviert wird, um Netzwerke in Subnetzwerke aufzuteilen. *Maske* kann als einzelne hexadezimale Nummer mit führendem 0x, also Internet-Adresse in Punkt-Notation, oder als in der Tabelle *networks* aufgeführtes Pseudo-Netzwerk angegeben werden.

pointopoint/-pointopoint [*Adresse*]
Ein- und Ausschalten von Point-to-Point-Verbindungen, so daß diese Verbindung der Kommunikation zwischen diesen beiden Rechnern vorbehalten ist.

up Schaltet eine Schnittstelle ein (zum Senden und Empfangen bereit).

trailers/-trailers
Ein- und Ausschalten einer nachfolgenden Kapselung beim Senden auf der Link-Ebene.

Adresse
Entweder ein Rechnername aus der Rechnerliste (*/etc/hosts*) oder eine Internet-Adresse in der Standard-Punkt-Notation.

inetd [*Option*] [*Konfigurationsdatei*]

TCP/IP-Befehl. Dämon für (fast) alle Internet-Dienste. **inetd** lauscht an mehreren Ports den einkommenden Verbindungsanfragen. Wenn eine solche eintrifft, startet **inetd** den passenden Server. Nachdem **inetd** gestartet ist, liest der Dämon zunächst seine Konfigurations-Information ein; entweder aus der angegebenen *Konfigurationsdatei* oder aus der Default-Konfigurationsdatei */etc/inetd.conf*. Dann ruft er **getservbyname** auf, legt für jeden Server ein Socket an und bindet jedes Socket an den Port des Servers. Danach wird für alle TCP-Sockets **listen** aufgerufen und mit **select** gewartet, bis eine Verbindungsanfrage oder ein Datagramm eintrifft.

Wenn auf einem überwachten Socket eine Verbindungsanfrage eintrifft, ruft **inetd accept** auf und legt ein neues Socket an. Dann startet es mit **fork**, **dup** und **exec** den passenden Server. **stdin**, **stdout** und **stderr** des neuen Servers sind auf das neue Socket gesetzt, womit der Server mit dem Prozeß des Clients verbunden ist.

Wenn an einem Datagramm-Socket Daten anliegen, startet **inetd** den passenden Server mit Hilfe von **fork**, **dup** und **exec**. Dem Server werden mögliche Argumente übergeben; **stdin**, **stdout** und **stderr** werden auf

→

inetd ←

das neue Socket gesetzt. Wenn ein Datagramm-Socket als **wait** markiert ist, muß der aufgerufene Server zunächst die Nachricht verarbeiten, bevor **inetd** das Socket für neue Verbindungen freigibt. Wenn das Socket auf **nowait** gesetzt ist, verarbeitet **inetd** auf diesem Port weitere eingehende Nachrichten.

inetd kann die folgenden Server starten: **bootpd**, **bootpgw**, **fingerd**, **ftpd**, **rexecd**, **rlogind**, **rshd**, **talkd**, **telnetd** und **tftpd**. **named**, **routed**, **rwhod**, **sendmail**, **listen** und alle NFS-Server sollten nie von **inetd** gestartet werden.

Wenn **inetd** das Signal **SIGHUP** empfängt, liest es seine Konfigurationsdatei neu ein. Dadurch können Dienste hinzugefügt, entfernt oder geändert werden, ohne daß **inetd** neu gestartet werden muß.

Option

-d
: Einschalten des Debuggings auf Socket-Ebene; Ausgeben der Debugging-Informationen nach **stdout**.

Dateien

/etc/inetd.conf
: Default-Konfigurationsdatei.

/var/run/inetd.pid
: Die Prozeß-ID von **inetd**.

---

init

init [*Option*] [*Runlevel*]

Option

-t *Sekunden*
: Beim Wechseln von Runleveln wird SIGKILL *Sekunden* Sekunden nach SIGTERM geschickt. Der Default ist 20.

Dateien

init ist der allererste Prozeß, den ein UNIX-Rechner beim Hochfahren startet. Er überprüft die Konsistenz der Dateisysteme und startet mit fork/exec andere Prozesse anhand der Angaben in */etc/inittab*. Welche Prozesse gestartet werden, ist abhängig vom *Runlevel*. Jedes Ende eines Prozesses wird in */var/run/utmp* und */var/log/wtmp* aufgezeichnet. Beim Wechsel des Runlevels schickt **init** an alle Prozesse, die im neuen Runlevel nicht laufen können, zunächst ein SIGTERM- und 20 Sekunden später ein SIGKILL-Signal.

Runlevel

Der aktuelle Runlevel kann mit **telinit** geändert werden. **telinit** ist oft nur ein Link auf **init**. Die Default-Runlevels unterscheiden sich von Distribution zu Distribution, aber die folgenden sind immer vorhanden:

0
: Anhalten des Systems.

1, s, S
: Single-User-Modus (Wartungsmodus).

6	Neustarten des Systems.	**init**
q, Q	Erneutes Einlesen von */etc/inittab*.	

Überprüfen Sie die Datei */etc/inittab*, um die auf Ihrem System vorgesehenen Runlevels zu erfahren.

---

insmod [*Optionen*] *Datei* [*Symbol*=*Wert* ...] — **insmod**

Laden des Moduls *Datei* in den Kernel, wobei alle auf der Kommandozeile definierten Symbole geändert werden. Wenn der Name der Moduldatei *Datei.o* oder *Datei.mod* ist, heißt das Modul *Datei*.

Optionen

-f
: Erzwingt das Laden des Moduls auch dann, wenn Probleme auftreten.

-m
: Gibt eine Ladetabelle aus.

-o *Name*
: Der Name des Moduls wird nicht aus dem Dateinamen abgeleitet, sondern ist *Name*.

-x
: Kein Export; es werden keine externen Symbole des Moduls zu der Symboltabelle des Kernels hinzugefügt.

---

install [*Optionen*] [*Datei*] *Verzeichnisse* — **install**

Wird hauptsächlich in Makefiles verwendet, um Dateien zu aktualisieren. **install** kopiert Dateien in vom Benutzer angegebene Verzeichnisse, wobei keine Dateien überschrieben werden. **install** ist ähnlich wie **cp**, versucht aber auch, Zugriffsrechte, Besitzer und Gruppe zu setzen.

-d, --**directory**
: Fehlende Verzeichnisse anlegen.

-g *group*, --**group** *Gruppe*
: Gruppen-ID der neuen Datei auf *Gruppe* setzen (kann nur vom Superuser ausgeführt werden).

-m *mode*, --**mode** *Modus*
: Zugriffsrechte auf die neue Datei auf *Modus* setzen. *Modus* kann oktal oder symbolisch angegeben werden. Der Default ist 0755.

-o [*owner*], --**owner** [*Besitzer*]
: Besitzer-ID auf *Besitzer* setzen; wenn kein Besitzer angegeben ist, auf **root** (kann nur vom Superuser ausgeführt werden).

-s, --**strip**
: Symboltabelle entfernen.

---

ipfw [*Optionen*] *Befehl*
ipfw [*Optionen*] *Aktion Antwort Protokoll Pfad*
ipfw [*Optionen*] *Aktion Kette* — **ipfw**

→

SysAdmin Befehle

ipfw
←

TCP/IP-Befehl. Überwachen der Account-Verwaltungs- und Firewallfunktionen. Sie können angeben, daß Pakete an bestimmte Ziele blockiert, weitergeleitet oder protokolliert werden.

Optionen

-n
: Nie IP-Adressen in Domänennamen auflösen.

-v
: Verbose-Modus. Ausgeben der Zähler für Pakete und Bytes.

Argumente

Aktion Kann eine von drei Aktionen sein: **entry**, **checking** oder **chain**. **entry** und **checking** benutzen die zweite oben aufgeführte Syntax, **chain** die dritte.

Kette Gibt die Kette an, auf der die Aktion ausgeführt werden soll. Siehe „Kettenaktionen" weiter unten.

Pfad Der *Pfad* wird immer in der Form **from** *Quelle* **to** *Ziel* angegeben. *Quelle* und *Ziel* können beide entweder Domänennamen oder IP-Adressen sein und dürfen Portnummern enthalten.

Protokoll
Einer der Werte **icmp**, **tcp**, **udp** oder **all**.

Antwort
Eine von drei möglichen Antworten: **accept** (Paket akzeptieren), **deny** (Paket ignorieren, keine Benachrichtigung schicken) oder **reject** (Paket zurückweisen, Zurücksenden einer „host unreachable"-Benachrichtigung.

Entry-Aktionen

a[dd]b[locking] *Antwort Protokoll Pfad*
Hinzufügen des *Antwort Protokoll Pfad*-Musters zur Musterliste in der blockierenden Firewall.

d[el]b[locking] *Antwort Protokoll Pfad*
Löschen des *Antwort Protokoll Pfad*-Musters von der Musterliste der blockierenden Firewall.

a[dd]f[orwarding] *Antwort Protokoll Pfad*
Hinzufügen des *Antwort Protokoll Pfad*-Musters zur Musterliste der weiterleitenden Firewall.

d[el]f[orwarding] *Antwort Protokoll Pfad*
Löschen des angegebenen Musters von der Musterliste der weiterleitenden Firewall.

a[dd]a[ccounting] *Antwort Protokoll Pfad*
Hinzufügen des angegebenen Musters zur Musterliste der kontoführenden Kette.

d[el]a[ccounting] *Antwort Protokoll Pfad*
Löschen des angegebenen Musters von der Musterliste der kontoführenden Kette.

Checking-Aktionen

ipfw

c[heck]b[locking] *Antwort Protokoll Pfad*
: Überprüft das angegebene Muster an der blockierenden Firewall.

c[heck]f[orwarding] *Antwort Protokoll Pfad*
: Überprüft das angegebene Muster an der weiterleitenden Firewall.

Chain-Aktionen

flush [firewall|accounting], f [f|a]
: Firewall-, Kontoführungs- oder beide Ketten löschen.

list [forwarding|blocking|accounting], l [f|b|a]
: Weiterleitende, blockierende, kontoführende oder alle Ketten löschen.

zero [accounting], zero [a]
: Alle Zähler der Kontoführung auf Null setzen.

---

ipfwadm -A|-B|-F *Befehl Parameter* [*Optionen*]

ipfwadm

Verwalten einer Firewall und ihrer Kontoführungsregeln. Es gibt drei Kategorien von Befehlen: Kontoführung, blockierende Firewall und weiterleitende Firewall.

Befehlskategorien

- **-A** Kontoführung
- **-B** Blockierende Firewall
- **-F** Weiterleitende Firewall

Befehle

Es muß jeweils genau ein Befehl angegeben werden. Dieser Befehl bezieht sich nur auf die angegebene Liste (Kontoführung, blokkierende Firewall oder weiterleitende Firewall).

-a [accept|deny|reject]
: Hinzufügen der angegebenen Parameter zur jeweiligen Liste. Wenn es sich bei der Liste um die blockierende oder weiterleitende Firewall handelt, muß auch eine Policy (**accept**, **deny** oder **reject**) angegeben werden.

-d [accept|deny|reject]
: Löschen der angegebenen Parameter von der jeweiligen Liste. Wenn es sich bei der Liste um die blockierende oder weiterleitende Firewall handelt, muß auch eine Policy (**accept**, **deny** oder **reject**) angegeben werden.

-l Alle Regeln der jeweiligen Liste ausgeben.

-z Alle Paket- und Bytezähler der angegebenen Liste zurücksetzen.

-f Alle Regeln von der angegebenen Liste entfernen.

-p [accept|deny|reject]
: Default-Policy der Liste angeben. Nicht möglich bei **-A**.

→

ipfwadm
←

-c Informationen über ein bestimmtes Paket ausgeben: Würde die angegebene Firewall das Paket akzeptieren (**accept**), es ohne Benachrichtigung ablehnen (**deny**) oder mit Benachrichtigung ablehnen (**reject**)? Nicht möglich bei -**A**.

Parameter

Die folgenden Parameter können nur im Zusammenhang mit -**a**, -**d** und -**c** verwendet werden.

-**P** [**tcp**|**udp**|**icmp**|**all**]
: Protokolltyp für die Operationen angeben.

-S *Adresse*[/*Maske*] [*Port* ...]
: Quelle angeben. *Maske* ist die Netzwerkmaske, z.B. 255.255.255.0.

-D *Adresse*[/*Maske*] [*Port* ...]
: Ziel angeben.

-I *Adresse*
: Pakete über die Schnittstelle *Adresse* empfangen.

Optionen

-b Bidirektionaler Modus: Die Befehle -**a** und -**d** beziehen sich auf IP-Pakete in beiden Richtungen.

-e Erweiterte Ausgabe: Wenn möglich, werden Schnittstellenadresse, Regeloptionen sowie Paket- und Bytezähler mit ausgegeben. Nur beim -l-Befehl möglich.

-n Alle Adressen und Ports in numerischer Form ausgeben.

-v Verbose-Modus: weitere Paketinformationen ausgeben. Nur möglich bei den Befehlen -**a**, -**d** und -**c**.

-x Expandierte Zahlendarstellung: Zahlen werden nicht gerundet, und **K** (1000) und **M** (1000K) werden nicht zur Abkürzung verwendet.

-y Bezieht sich auf Pakete, bei denen das SYN-Bit, aber nicht das ACK-Bit gesetzt ist. Nur möglich bei den Befehlen -**a** und -**d**.

kbdrate

kbdrate [*Optionen*]

Einstellen der Geschwindigkeit und Verzögerung beim Wiederholen von Zeichen durch die Tastatur. Wenn keine Optionen angegeben werden, wird die Wiederholrate auf 10,9 Zeichen pro Sekunde und die Default-Verzögerung auf 250 Millisekunden eingestellt. Beim Hochfahren von Linux wird die Wiederholrate dagegen auf 30 Zeichen pro Sekunde gesetzt.

Optionen

-s Keine Meldungen ausgeben.

-r *Rate*
: Angeben der Wiederholrate. Die folgenden Werte sind erlaubt (jeweils in Zeichen pro Sekunde): 2.0, 2.1, 2.3, 2.5, 2.7, 3.0, 3.3, 3.7, 4.0, 4.3, 4.6, 5.0, 5.5, 6.0, 6.7, 7.5, 8.0,

8.6, 9.2, 10.0, 10.9, 12.0, 13.3, 15.0, 16.0, 17.1, 18.5, 20.0, 21.8, 24.0, 26.7 und 30.0.

-d *Verzögerung*
  Angeben der Verzögerung. Die folgenden Werte (in Millisekunden) sind erlaubt: 250mS, 500mS, 750mS und 1000mS.

kbdrate

kerneld

kerneld

kerneld lädt Kernelmodule automatisch bei Bedarf, wodurch der Speicherverbrauch durch den Kernel aufgrund unbenutzter Module reduziert und das manuelle Laden von Modulen mit **modprobe** oder **insmod** unnötig gemacht wird. Wenn ein Modul mehr als eine Minute nicht verwendet worden ist, wird es von **kerneld** automatisch entfernt.

kerneld ist ein Bestandteil des modutils-Pakets und wird während der Konfiguration des Kernels eingerichtet. Seine Funktionalität wird durch Interaktionen zwischen diesem Paket und dem Kernel gewährleistet. **kerneld** kennt die gebräuchlichsten Modultypen. Wenn mehr als ein Modul für ein Gerät (z.B. einen Netzwerktreiber) verwendet werden kann, zieht **kerneld** die Konfigurationsinformationen in */etc/conf. modules*, die Pfadinformationen und Aliasnamen für alle ladbaren Module enthalten, zu Rate, um das richtige Modul zu bestimmen.

kerneld kann auch dazu verwendet werden, um Netzwerkverbindungen über Telefonleitungen wie z.B. SLIP oder PPP bei Bedarf aufzubauen („dial-on-demand"). Der Wunsch nach einer Netzwerkverbindung veranlaßt **kerneld**, die entsprechenden Module zu laden und die Verbindung zum Server aufzubauen.

klogd [*Optionen*]

klogd

Bestimmt, welche Kernelmeldungen auf der Konsole ausgegeben werden sollen, ordnet die Meldungen nach Prioritäten und speichert sie via **syslogd** ab. Auf vielen anderen Betriebssystemen sind **syslogd** und **klogd** ein Programm, aber auf Linux sind die Aufgaben getrennt. **klogd** erhält die Kernelmeldungen durch Absuchen des */proc*-Dateisystems sowie von Systemaufrufen von **syslogd**. Defaultmäßig erscheinen keine Meldungen auf der Konsole. Die Meldungen sind in acht Ebenen aufgeteilt (0–7); die Ebenennummer wird jeder Meldung vorangestellt.

Prioritäts-Ebenen

0 Notfallsituation (**KERN_EMERG**).

1 Ein fataler Fehler ist aufgetreten (**KERN_ALERT**).

2 Ein schwerwiegender Fehler ist aufgetreten (**KERN_CRIT**).

3 Ein Fehler ist aufgetreten (**KERN_ERR**).

4 Warnung (**KERN_WARNING**).

5 Normale Situation, eine Überprüfung ist aber angeraten (**KERN_NOTICE**).

→

SysAdmin Befehle

klogd ←

6 Information (KERN_INFO).

7 Debugging-Meldung (KERN_DEBUG).

Optionen

-c *Ebene*
: Alle Meldungen mit einer höheren Priorität (niedrigeren Nummer) als *Ebene* werden auf die Konsole ausgegeben.

-d
: Debugging-Modus.

-f *Datei*
: Alle Meldungen in *Datei* ausgeben; normales Loggen unterdrücken.

-k *Datei*
: *Datei* als Quelle für Kernelsymbole verwenden.

-n
: Nicht automatisch in den Hintergrund gehen. Dies wird benötigt, wenn **klogd** von **init** aus gestartet wird.

-o
: One-Shot-Modus. Alle aktuellen Meldungen werden nach Prioritäten sortiert und ausgegeben, danach beendet sich **klogd**.

-s
: Keine Meldungen aus dem */proc*-Dateisystem lesen.

Dateien

/usr/include/linux/kernel.h, */usr/include/sys/syslog.h*
: Definitionen der einzelnen Ebenen.

/proc/kmsg
: Die Datei in */proc*, die nach Meldungen abgesucht wird.

/var/run/klogd.pid
: Die Prozeß-ID von **klogd**.

---

ksyms

ksyms [*Optionen*]

Ausgeben einer Liste aller exportierten Kernelsymbole mit Namen, Adresse und definierendem Modul (wo möglich).

Optionen

-a
: Auch Symbole nicht geladener Module ausgeben.

-h
: Startmeldung nicht mit ausgeben.

-m
: Startadresse und Größe mit ausgeben. Nur sinnvoll für Symbole aus geladenen Modulen.

Dateien

/proc/ksyms
: Enthält die gleiche Information.

---

ldconfig

ldconfig [*Optionen*] *Verzeichnisse*

Untersucht die Bibliotheken in *Verzeichnisse*, */etc/ld.so.conf*, */usr/lib* und */lib* und aktualisiert Links und den Cache, soweit notwendig. **ldconfig** wird normalerweise aus Startdateien heraus oder nach der Installation neuer dynamischer Bibliotheken ausgeführt.

Optionen

ldconfig

-D
: Debugging. Alle normalen Funktionen werden nicht ausgeführt.

-l
: Bibliotheks-Modus. Erwartet, daß die Argumente keine Verzeichnisse, sondern Bibliotheken sind, und legt die Links der angegebenen Bibliotheken automatisch an.

-n
: Durchsuchung von */usr/lib* und */lib* sowie Einlesen von */etc/ld.so.conf* unterdrücken; den Cache nicht verwenden.

-N
: Den Cache nicht verwenden, nur die Links anlegen.

-p
: Alle Verzeichnisse und möglichen Bibliotheken im Cache ausgeben. Erwartet keine Argumente.

-v
: Verbose-Modus. Ausgeben der Versionsnummer und jedes Verzeichnisses einschließlich der dort angelegten Links.

-X
: Keine Links anlegen, nur den Cache neu aufbauen.

Dateien

/lib/ld.so
: Linker- und Ladeprogramm.

/etc/ld.so.conf
: Liste der Verzeichnisse, in denen üblicherweise Bibliotheken abgelegt werden.

/etc/ld.so.cache
: Liste der Bibliotheken, die in den in */etc/ld.so.conf* angegebenen Verzeichnissen gefunden wurden.

---

logger [*Optionen*] [*Nachricht* . . .]

logger

TCP/IP-Befehl. Hinzufügen von Meldungen in die Log-Dateien des Systems. Die Meldung kann von der Kommandozeile oder der Standardeingabe gelesen werden.

Optionen

-f *Datei*
: Einlesen der *Nachricht* aus *Datei*.

-i
: Hinzufügen der Prozeßnummer des **logger**-Prozesses.

-p *Pri*
: Meldung mit der angegebenen Priorität *Pri* versehen. Der Default ist „**user.notice**".

-t *Tag*
: Jede Zeile mit dem angegebenen *Tag* markieren.

---

login [*Name* | *Option*]

login

Am System anmelden. **login** erfragt den Benutzernamen (*Name*), der auch auf der Kommandozeile angegeben werden kann, und das Paßwort (wo nötig).

Wenn das Anmelden erfolgreich war, aktualisiert **login** Buchführungsdateien, setzt verschiedene Umgebungsvariablen, teilt dem Benutzer mit, ob neue Email für ihn vorliegt, und führt die Start-Shellskripten aus.

→

SysAdmin Befehle

login ←

Wenn die Datei */etc/nologin* existiert, kann sich außer **root** kein Benutzer anmelden. Der Inhalt dieser Datei wird vor Abbruch der Verbindung angezeigt.

root kann sich nur auf Terminals anmelden, die in der Datei */etc/securetty* eingetragen sind. Wenn die Datei *~/.hushlogin* existiert, wird ein stilles Anmelden durchgeführt. Wenn die Datei */var/adm/lastlog* existiert, wird die Zeit des letzten Anmeldens ausgegeben.

Optionen

-f
: Unterdrücken der zweiten Anmelde-Authentisierung.

-h *Rechner*
: Angeben des Namens eines entfernten Rechners. Wird normalerweise von Servern und nicht von Menschen benutzt und kann nur von **root** ausgeführt werden.

-p
: Vorige Umgebung erhalten.

lpd

lpd [*Option*] [*Port*]

TCP/IP-Befehl. Drucker-Dämon. **lpd** wird üblicherweise beim Hochfahren des Systems gestartet. Der Dämon liest die Datei */etc/printcap*, um herauszufinden, welche Drucker existieren, und druckt eventuell noch vor einem Systemabsturz liegengebliebene Dateien aus. Danach akzeptiert er Anforderungen, Druckdateien in eine Warteschlange zu übertragen, kopiert Dateien in den Spool-Bereich, zeigt den Zustand der Warteschlange an oder entfernt Druckaufträge aus der Warteschlange. In allen Fällen wird für jede Anfrage ein neuer Prozeß gestartet, während der Dämon selbst weiter auf Anfragen wartet. Wenn *Port* angegeben wird, lauscht **lpd** an diesem Port, anderenfalls verwendet er **getservbyname**, um den richtigen Port herauszufinden.

Die Datei *lock* in jedem Spool-Verzeichnis verhindert, daß mehrere Dämonen gleichzeitig aktiv werden können. Nachdem ein Dämon diese Sperre gesetzt hat, durchsucht er das Verzeichnis nach Dateien, deren Namen mit **cf** anfangen. Diese Dateien enthalten Namen von Dateien, die ausgedruckt werden sollen, sowie weitere Anweisungen. Jede Zeile beginnt mit einem Schlüsselbuchstaben, der angibt, wie der Rest der Zeile zu interpretieren ist. Die folgenden Schlüsselbuchstaben sind bekannt:

C
: Klassifizierung — ein String, der in der Klassifizierungszeile des Zwischenblattes ausgegeben wird

c
: **cifplot**-Datei

f
: Eine formatierte Datei, d.h. eine zu druckende Datei, die bereits formatiert worden ist

g
: Graphendatei

l
: Eine formatierte Datei, bei der die Ausgabe von Seitenvorschüben und der Ausdruck von Steuerzeichen unterdrückt werden sollen

H
: Rechnername — der Name des Rechners, auf dem **lpd** gestartet wurde

J Jobname — String, der als Jobname auf dem Zwischenblatt verwendet wird
L Literal — Diese Zeile enthält Identifizierungsinformationen aus der Paßwortdatei und veranlaßt, daß eine Titelseite gedruckt wird
M Senden einer Mail an den angegebenen Benutzer, wenn der Druckjob vollständig ausgeführt worden ist
n **ditroff**-Datei
P Der Name des Benutzers, der **lpd** aufgerufen hat
r DVI-Datei
T Ein String, der als Titel für den **pr**-Befehl verwendet wird
t **troff**-Datei
U Der Name der Datei, die nach dem Ausdruck entfernt werden soll

Option

-l Alle gültigen Anfragen protokollieren

Dateien

/etc/printcap Beschreibungsdatei für Drucker

*/var/spool/** Spool-Verzeichnisse

/var/spool//minfree*
Platz, der im Spool-Bereich mindestens freibleiben muß

*/dev/lp** Drucker-Gerätedateien

/etc/hosts.equiv
Namen der Rechner, die auf den Drucker zugreifen dürfen

/etc/hosts.lpd Namen der Rechner, die auf den Drucker zwar zugreifen dürfen, aber keine administrativen Befehle ausführen können.

lsmod

Anzeigen aller geladenen Module mit Namen, Größe (in 4k-Einheiten) und einer Liste der Module, die diese benötigen (falls zutreffend).

Dateien

/proc/modules
Enthält die gleiche Information.

makedbm [*Optionen*] *Eingabedatei Ausgabedatei*

NFS/NIS-Befehl. Erzeugt eine **dbm**-Datei für NIS. **makedbm** erzeugt aus der *Eingabedatei* ein Dateienpaar *Ausgabedatei.pag* und *Ausgabedatei.dir* im **ndbm**-Format. Jede Zeile der Eingabedatei wird in einen **dbm**-Datensatz konvertiert. Alle Zeichen bis zum ersten Tabulator oder Leerzeichen bilden den Schlüssel, alle anderen die Daten. Wenn eine Zeile mit \& endet, werden die Daten dieser Zeile auf der nächsten Zeile fortgesetzt. Das #-Zeichen muß von den NIS-Clients interpretiert werden, **makedbm** betrachtet es nicht als Kommentarzeichen. Die

→

Eingabedatei kann auch - sein; in diesem Fall wird von der Standardeingabe gelesen.

makedbm erzeugt einen speziellen Eintrag mit dem Schlüssel **yp_LAST_MODIFIED**, wobei es sich beim Wert um den Zeitstempel der Eingabedatei (oder der aktuellen Datei, falls die Eingabe von der Standardeingabe gelesen wird) handelt.

Optionen

-b
: Die Tabelle wird an alle Server mit Hilfe des Name-Server-Dämons **named** weitergegeben.

-d yp_DOMAIN_NAME
: Anlegen eines speziellen Eintrags mit dem Schlüssel **yp_DOMAIN_NAME**.

-i yp_INPUT_FILE
: Anlegen eines speziellen Eintrags mit dem Schlüssel **yp_INPUT_FILE**.

-l
: Schlüssel der angegebenen Tabelle in Kleinbuchstaben konvertieren.

-m yp_MASTER_NAME
: Anlegen eines speziellen Eintrags mit dem Schlüssel **yp_MASTER_NAME**. Wenn kein Master-Rechnername angegeben wird, wird der Wert von **yp_MASTER_NAME** auf den lokalen Rechnernamen gesetzt.

-o yp_OUTPUT_FILE
: Anlegen eines speziellen Eintrags mit dem Schlüssel **yp_OUTPUT_FILE**.

-s
: Sichere Tabellen. Es werden nur Verbindungen aus sicheren NIS-Netzwerken akzeptiert.

-u dbm *Dateiname*
: Zurückkonvertieren einer **dbm**-Datei — Ausgeben einer **dbm**-Datei mit einem Eintrag pro Zeile und je einem Leerzeichen zwischen Schlüsseln und Werten.

Beispiel

Es ist nicht schwer, Shell-Skripten zu schreiben, um Standarddateien wie */etc/passwd* in die Schlüssel/Wert-Form zu konvertieren, die von **makedbm** verwendet wird. Beispielsweise liest das folgende **awk**-Programm:

```
BEGIN { FS =":";OFS = "\t";}
{ print $1, $0}
```

die Datei */etc/passwd* und konvertiert sie in eine Form, die von **makedbm** in die NIS-Datei *passwd.byname* umgewandelt werden kann. Der Schlüssel ist dabei der Benutzername und der Wert der Rest der Zeile in */etc/passwd*.

makemap [*Optionen*] *Typ Name*

makemap

Konvertiert Daten von der Standardeingabe in **sendmail**-Datenbanktabellen. Die Eingabe sollte in der Form

Schlüssel Wert

vorliegen. Kommentarzeilen können mit # eingefügt und Parameter mit %**n** substituiert werden. Ein wörtliches % muß als %% eingegeben werden. Der Typ ist entweder **dbm**, **hash** oder **btree**. *Name* ist ein Dateiname, an den **makemap** die Standardsuffixe anhängt.

Optionen

-f
: Umsetzung von Groß- in Kleinbuchstaben nicht vornehmen.

-N
: An jeden Schlüssel ein Nullbyte anhängen.

-o
: An eine existierende Datei anhängen, anstatt sie zu ersetzen.

-r
: Wenn Schlüssel schon existieren, werden sie ersetzt. Defaultmäßig beendet sich **makemap**, wenn es auf doppelte Schlüssel trifft.

-v
: Verbose-Modus.

mke2fs [*Optionen*] *Gerät* [*Blöcke*]
mkfs.ext2 [*Optionen*] *Gerät* [*Blöcke*]

mke2fs

Das *Gerät* als Linux Second Extended Dateisystem formatieren. Die Anzahl der Blöcke auf dem Gerät kann angegeben oder von **mke2fs** erraten werden.

Optionen

-b *Blockgröße*
: Blockgröße in Byte angeben.

-c
: Vor der Formatierung das Gerät auf defekte Blöcke überprüfen.

-f *Fragmentgröße*
: Fragmentgröße in Byte angeben.

-i *Byte-pro-Inode*
: Anlegen einer Inode für je *Byte-pro-Inode* Byte Plattenplatz. Der Wert für *Byte-pro-Inode* muß 1024 oder größer sein, der Default ist 4096.

-l *Dateiname*
: Eine Liste mit defekten Blöcken aus *Dateiname* lesen.

-m *Reserve*
: *Reserve* Prozent der Blöcke für den Superuser reservieren.

-q
: Stillmodus.

-v
: Verbose-Modus.

-S
: Nur den Superblock und die Gruppendeskriptoren schreiben und das Schreiben der Inode-Tabelle sowie der Block- und Inode-Bitmaps unterdrücken. Nur nützlich, wenn versucht wird, beschädigte Dateisysteme zu retten.

mkfs

mkfs [*Optionen*] [*Dateisystem–Optionen*] *Dateisystem* [*Blöcke*]

Anlegen eines Dateisystems auf einem Gerät (z.B. einer Festplattenpartition). *Dateisystem* ist entweder der Name eines Geräts oder der Einhängepunkt. **mkfs** ist eigentlich nur ein Zugangsprogramm, das die dateisystem-spezifische Version von **mkfs** abhängig vom Dateisystemtyp, der mit der -t-Option angegeben wird, aufruft. Zum Anlegen eines Linux-Second-Extended- Dateisystems wird beispielsweise **mkfs.ext2** (was wiederum identisch mit **mke2fs** ist) aufgerufen. MS-DOS-Dateisysteme werden mit **mkfs.msdos** angelegt. *Dateisystem-Optionen* sind Optionen, die spezifisch für das jeweilige Dateisystem sind. *Blöcke* ist die Größe des Dateisystems in 1024-Byte- Blöcken.

Optionen

-V
: Ausführliche Ausgaben u.a. aller Befehle, die zum Anlegen des jeweiligen Dateisystems notwendig sind.

-t *Dateisystemtyp*
: Teilt **mkfs** mit, was für ein Dateisystem angelegt werden soll.

Dateisystem-spezifische Optionen
: Diese Optionen müssen den generischen Optionen folgen und dürfen nicht mit diesen vermischt werden. Die meisten Formatierprogramme für Dateisysteme unterstützen die folgenden drei Optionen:

-c
: Vor Anlegen des Dateisystems das Gerät auf defekte Blöcke überprüfen.

-l *Datei*
: Aus der *Datei* eine Liste der defekten Blöcke des Geräts einlesen.

-v
: Ausführliche Ausgaben machen.

mkfs.minix

mkfs.minix [*Optionen*] *Gerät Größe*

Anlegen eines MINIX-Dateisystems auf *Gerät*.

mklost+found

mklost+found

Anlegen eines *lost+found*-Verzeichnisses im aktuellen Arbeitsverzeichnis. Gedacht für Linux-Second-Extended-Dateisysteme.

mkswap

mkswap [*Option*] *Gerät* [*Größe*]

Anlegen eines Auslagerungsbereichs (swap space) auf dem angegebenen *Gerät*. Die Größe wird in Blöcken von etwa 4K angegeben.

Option

-c
: Suchen nach defekten Blöcken, bevor der Auslagerungsbereich angelegt wird.

modprobe [*Optionen*] [*Module*]

Ohne Optionen versucht **modprobe**, das angegebene Modul und alle anderen Module, von denen es abhängig ist, zu laden. Wenn mehr als ein Modul angegeben wird, wird nur versucht, die weiter hinten stehenden Module zu laden, wenn die weiter vorn stehenden nicht geladen werden konnten.

Optionen

-a
Alle angegebenen Module laden, nicht nur das erste.

-l [*Muster*]
Alle vorhandenen Module anzeigen. Diese Option kann mit -t kombiniert werden, um den Typ der Module anzugeben. Wenn *Muster* angegeben wird, wird danach gesucht.

-r
Die angegebenen Module und alle, auf denen sie aufbauen, entfernen.

-t *Typ*
Nur einen bestimmten Modultyp laden. In */etc/modules.conf* finden Sie die Verzeichnisse, die jeweils alle Module eines Typs enthalten.

Dateien

/etc/conf.modules
Informationen über Module: Auf welchen anderen Modulen sie aufbauen und welche Verzeichnisse bestimmte Modultypen enthalten.

/sbin/insmod, /sbin/rmmod, /sbin/depmod
Programme, die **modprobe** benutzt.

mount [*Optionen*] [*Gerätedatei*] [*Einhängepunkt*]

Einhängen einer Dateistruktur. Mit **mount** wird dem System mitgeteilt, daß eine entfernbare Dateistruktur an der *Gerätedatei* vorliegt. Die Dateistruktur wird am *Einhängepunkt* eingehängt. Es muß sich dabei um ein bereits existierendes Verzeichnis handeln, das leer sein sollte. Dieser Einhängepunkt wird die Wurzel der gerade eingehängten Dateistruktur. Wenn **mount** ohne Argumente aufgerufen wird, gibt es den Namen jedes eingehängten Geräts, den zugehörigen Einhängepunkt, den Zeitpunkt des Einhängens und die Information, ob die Dateistruktur schreibbar ist, aus. Nur der Superuser kann **mount** verwenden.

Optionen

-a
Alle in */etc/fstab* genannten Dateisysteme einhängen. Dies ist die einzige Option, bei der keine *Gerätedatei* und kein *Einhängepunkt* angegeben werden können.

-f
Vorgetäuschtes Einhängen. Gerät und Verzeichnis werden überprüft, das eigentliche Einhängen aber nicht vorgenommen.

-n
Das Einhängen nicht in */etc/mtab* verzeichnen.

→

- o *Option*

  Dies ist die einzige Option von **mount**, die eine **Gerätedatei** und einen *Einhängepunkt* erwartet. Sie können den Einhängepunkt näher mit folgenden *Optionen* spezifizieren:

  async Ein- und Ausgaben vom und zum Gerät asynchron lesen.

  auto Einhängen mit der **-a**-Option erlauben.

  defaults Bei allen Optionen werden die Defaultwerte verwendet (**async**, **auto**, **dev**, **exec**, **nouser**, **rw** und **suid**).

  dev Alle Gerätedateien, die sich im Dateisystem befinden, interpretieren.

  exec Binärdateien dürfen ausgeführt werden.

  noauto Einhängen über die **-a**-Option ist nicht erlaubt.

  nodev Gerätedateien, die sich im Dateisystem befinden, nicht interpretieren.

  noexec Binärdateien dürfen nicht ausgeführt werden.

  nosuid **suid** und **sgid**-Bits werden nicht berücksichtigt.

  nouser Nur der Superuser hat Zugriff auf das Dateisystem.

  remount Erwartet, daß das Dateisystem bereits eingehängt ist, und hängt es erneut ein.

  ro Nur lesender Zugriff auf das Dateisystem.

  rw Lesender und schreibender Zugriff auf das Dateisystem.

  suid Berücksichtigen der **suid**- und **sgid**-Bits.

  sync Ein- und Ausgaben von und zu dem Gerät synchron lesen.

  user Auch andere Benutzer als der Superuser können das Dateisystem einhängen. In diesem Fall sind die Defaults **nodev**, **noexec** und **nosuid**, wenn keine anderen Angaben gemacht werden.

  check=relaxed|normal|strict
  Gibt an, wie streng die Integration eines MS-DOS-Dateisystems gehandhabt werden soll.

  conv=binary|text|auto
  Gibt die Methode an, mit der MS-DOS- und ISO-9660-Dateisysteme konvertiert werden sollen.

  debug Debugging für MS-DOS- und **ext2fs**-Dateisysteme einschalten.

  errors=continue|remount|ro|panic
  Angeben, welche Aktion beim Auftreten eines Fehlers ausgeführt werden soll. Nur für das **ext2fs**-Dateisystem.

- v Ausführliche Ausgaben über den Einhängevorgang machen.

- w Dateisystem zum Lesen und Schreiben einhängen.

Dateien

/etc/mtab

Eine Liste der aktuell eingehängten Dateien und der Optionen, mit denen sie eingehängt worden sind.

rpc.mountd [*Optionen*]

mountd

NFS/NIS-Befehl. NFS Mount Server. **mountd** liest die Datei */etc/exports*, um zu bestimmen, welche Dateisysteme anderen Rechnern zum Einhängen zur Verfügung stehen sollen. Außerdem stellt er Informationen zur Verfügung, welcher Client welches Dateisystem eingehängt hat. Siehe auch **nfsd**.

Optionen

-d, --**debug**
: Debugging-Modus. Alle Debugging-Informationen werden über **syslogd** ausgegeben.

-f *Datei*, --**exports-file** *Datei*
: Die Export-Zugriffsrechte werden aus *Datei* anstelle von */etc/exports* gelesen.

-n, --**allow-non-root**
: Erlaubt auch Einhänge-Anforderungen, die auf einem nichtreservierten Port eintreffen.

-p, --**promiscuous**
: Erlaubt Anforderungen von jedem Rechner.

-r, --**re-export**
: Erlaubt das Weiterexportieren importierter Dateisysteme.

-v, --**version**
: Ausgeben der Versionsnummer.

Dateien

/etc/exports
: Informationen über Zugriffsrechte zum Einhängen.

named [*Optionen*]

named

TCP/IP-Befehl. Internet Domain Name Server. **named** wird von Resolver-Bibliotheken benutzt, um Zugriff auf die verteilte Namensdatenbank des Internets zu erhalten. Ohne Argumente gestartet, liest **named** seine Anfangsdaten aus der Datei */etc/named.boot* und lauscht an einem privilegierten Port den Anfragen. Siehe RFC 1034 und RFC 1035, um weitere Informationen zu erhalten.

Es gibt mehrere verschiedene **named**-Programme in den verschiedenen Linux-Archiven, die alle unterschiedliches Verhalten aufweisen. Seien Sie also nicht beunruhigt, wenn Ihre Version sich nicht so verhält wie hier beschrieben. Es sollte eine Dokumentation dazu vorliegen.

Optionen

-d *Debugebene*
: Ausgeben von Debugging-Informationen. *Debugebene* gibt an, welche Informationen ausgegeben werden sollen.

-p *Port*
: Verwenden von Port-Nummer *Port*. Der Default ist 53.

→

SysAdmin Befehle

named
←

[-b] *Bootdatei*
Die Datei, die anstelle von *named.boot* verwendet werden soll. Das **-b** ist optional und erlaubt es, einen Dateinamen anzugeben, der mit einem Querstrich beginnt.

Dateien

/etc/named.boot
Wird beim Starten von **named** gelesen.

netdate

netdate [*Optionen*] [*Protokoll*] *Rechnernamen* ...

TCP/IP-Befehl. Stellt die Uhr des Systems nach der Uhrzeit eines der Rechner der *Rechnernamen*-Liste. **netdate** versucht herauszufinden, welche Rechner die zuverlässigste Quelle ist. Wenn **netdate** nicht vom Superuser gestartet wird, teilt es lediglich die ermittelte Uhrzeit mit, stellt aber die Systemzeit nicht ein. Sie können ein *Protokoll* zur Datenübertragung angeben, entweder einmal für alle oder auch für jeden Rechner einzeln. Es stehen zur Verfügung: **udp** (der Default) und **tcp**.

-l *Zeit* Der zuverlässigste Rechner von der Liste wird folgendermaßen bestimmt: Die Rechner werden nach den Zeiten, die sie zurückmelden, in Gruppen sortiert. Der erste Rechner der größten Gruppe wird ein zweites Mal nach der Zeit gefragt. Bei beiden Abfragen wird die Differenz zwischen der zurückgegebenen Zeit und der aktuellen Systemzeit des lokalen Rechners gespeichert. Diese beiden Differenzen werden verglichen. Wenn der Unterschied größer als *Zeit* ist (der Default ist fünf Sekunden), wird der Rechner als inakkurat angesehen.

-v Zeigt die Gruppen an, nach denen die Rechner sortiert sind.

netstat

netstat [*Optionen*]

TCP/IP-Befehl. Zeigt den Netzwerkstatus an. Für jedes aktive Socket werden das Protokoll, die Anzahl der Bytes, die zum Empfang und zum Versand anstehen, die Portnummer, die Adresse und Portnummer der Gegenstelle sowie der Zustand des Sockets ausgegeben.

Optionen

-a Zeigt den Zustand aller Sockets an, nicht nur der aktiven.

-c Ständiges Anzeigen der Informationen, die Anzeige wird einmal pro Sekunde aktualisiert.

-i Statistiken über Netzwerkgeräte mit anzeigen.

-n Netzwerkadressen als Nummern anzeigen.

-o Zusätzliche Informationen über Timerzustände ausgeben.

-r Routing-Tabellen anzeigen.

-t Nur TCP-Sockets anzeigen.

-u Nur UDP-Sockets anzeigen.

-v Versionsnummer ausgeben und Programm beenden.

-w Nur reine IP-Sockets (weder UDP noch TCP) anzeigen.

-x Nur UNIX-Domänensockets anzeigen.

netstat

rpc.nfsd [*Optionen*]

nfsd

rpc.nfsd ist ein Dämon, der bei Bedarf die NFS-Serverdämonen startet, welche die Anfragen von Clients bedienen. Diese Dämonen befinden sich im Benutzeradreßraum. Die Optionen sind die gleichen wie bei **mountd**.

nslookup [–*Option* ...] [*zu_findender_Rechner* | – [*Server*]]

nslookup

TCP/IP-Befehl. Stellt Anfragen an Internet Domain Name Server. **nslookup** kennt zwei Modi: interaktiv und nicht-interaktiv. Im interaktiven Modus kann der Benutzer Name Server nach Informationen über diverse Rechner und Domänen befragen oder eine Liste aller Rechner einer Domäne ausgeben. **nslookup** startet im interaktiven Modus, wenn entweder kein Argument angegeben wird oder das erste Argument ein Bindestrich und das zweite Argument ein Rechnername oder die Internet-Adresse eines Name Servers ist. Im nicht-interaktiven Modus werden nur der Name und die angeforderte Information über einen Rechner oder eine Domäne ausgegeben. Der nicht-interaktive Modus wird gestartet, wenn das erste Argument der Name eines Rechners ist, zu dem Informationen angefordert werden. Das optionale zweite Argument gibt den zu verwendenden Name Server an.

Optionen

Alle beim interaktiven **set**-Befehl aufgeführten Optionen können mit der Syntax -*Schlüsselwort*[=*Wert*] auch von der Kommandozeile aus verwendet werden, indem dem Schlüsselwort ein – vorangestellt wird.

Interaktive Befehle

exit **nslookup** beenden.

finger [*Name*] [>|>>*Datei*]
: Mit dem Finger-Server auf dem aktuellen Rechner verbinden und dabei *Datei* anlegen oder die Ausgabe an diese Datei anhängen.

help, ?
: Ausgeben einer kurzen Befehlsübersicht.

host [*Server*]
: Informationen über *host* vom aktuellen Defaultserver oder dem angegebenen *Server* abfragen.

ls -[ahd] *Domäne* [>|>>*Datei*]
: Ausgeben der über *Domäne* verfügbaren Informationen. Diese können auch in eine neu angelegte Datei geschrieben oder an eine bereits vorhandene angehängt werden. Mit der **-a**-Option werden Aliasnamen der Rechner der Domäne ausgegeben, mit

SysAdmin Befehle

→

-h Informationen über die CPUs und Betriebssysteme der Rechner und mit -d der gesamte Inhalt eines Zonentransfers.

lserver *Domäne*
Wechseln des Defaultservers zu *Domäne*. Der initiale Server wird verwendet, um die Informationen über *Domäne* zu erhalten.

root Wechseln des Defaultservers zum Server der Wurzeldomäne des gesamten Namensraums.

server *Domäne*
Wechseln des Defaultservers zu *Domäne*. Der aktuelle Defaultserver wird verwendet, um die Informationen über *Domäne* zu erhalten.

set *Schlüsselwort*[=*Wert*]
Ändern von Einstellungen für Anfragen. Die möglichen Schlüsselwörter sind:

all Ausgeben der aktuellen Werte der häufig benutzten **set**-Optionen.

class=*Name*
Einstellen der Abfragenklasse auf **IN** (Internet), **CHAOS**, **HESIOD** oder **ANY** (alle). Der Default ist IN.

domain=*Name*
Wechseln des Default-Domänennamens zu *Name*.

[no]debug
Debugging-Modus ein- oder ausschalten.

[no]d2 Ausführlichen Debugging-Modus ein- oder ausschalten.

[no]defname
Den Default-Domänennamen an jede Anfrage anhängen.

[no]ignoretc
Bei verkürzten Paketen eine erneute Anfrage, diesmal mit einer **tcp**-Verbindung, starten (**noignoretc**) oder einen Fehler ausgeben (**ignoretc**).

[no]recurse
Teilt dem Name Server mit, ob er andere Name Server zu Rate ziehen soll oder nicht, wenn er die gewünschte Information nicht selbst bereitstellen kann.

[no]search
Wenn **search** eingeschaltet ist und der zu suchende Name nicht mit einem Punkt endet, werden die Domänen der Suchliste (**srchlist**) an den Namen angehängt; wenn **search** ausgeschaltet ist, der Name der Defaultdomäne.

[no]vc
Schaltet die Verwendung von TCP statt UDP ein bzw. aus.

port=*Port*
Verwenden von Port-Nummer *Port* beim Name Server.

querytype= *Wert*
Siehe **type=***Wert*.

retry=*Anzahl*
Gibt die Anzahl der Wiederholversuche an.

root=*Rechner*
Stellt den Wurzelserver für den **root**-Befehl ein.

srchlist=*Domänen*
Die angegebenen *Domänen* werden durchsucht.

timeout=*Sekunden*
Ändert das Timeout-Intervall für das Warten auf eine Antwort auf *Sekunden* Sekunden.

type=*Wert*
Einstellen des Informationstyps, der auf eine Anfrage hin zurückgegeben werden soll. Mögliche Werte:

A	Internet-Adresse des Rechners
ANY	Alle verfügbaren Informationen
CNAME	Kanonischer Name eines Aliasnamens
HINFO	CPU und Betriebssystemtyp
MD	Ziel einer Email (veraltet)
MG	Gruppenmitgliedschaft einer Email (experimentell)
MINFO	Mailbox- oder Mailinglisteninformation (experimentell)
MR	Tatsächlicher Name einer Mailbox (experimentell)
MX	Weiterleitungsinformationen für Email
NS	Name-Server der benannten Zone
PTR	Rechnername oder ein Zeiger auf weitere Informationen
SOA	Kontaktinformationen einer Zone (*start-of-authority*)
TXT	Textuelle Information
UINFO	Benutzerinformation
WKS	Unterstützte, allgemein bekannte Dienste

view *Dateiname*
Sortieren und Anzeigen der Ausgaben vorhergehender **ls**-Befehle mit **more**.

/usr/sbin/rpc.pcnfsd

pcnfsd

NFS/NIS-Befehl. NFS-Authentisierungs- und Drucker-Server. **pcnfsd** ist ein RPC-Server, der ONC-Clients auf PCs unterstützt. Er wird häufig aus */etc/rc.d/rc.inet2* gestartet. **pcnfsd** liest zunächst die Konfigurationsdatei */etc/pcnfsd.conf* ein und bedient danach Anfragen, die an die Programmnummer 150001 gerichtet sind. Die aktuelle Version des **pcnfsd**-Dämonens unterstützt (zum Zeitpunkt der Drucklegung dieses Buches) sowohl Version 1 als auch Version 2 des **pcnfs**-Protokolls. Anfragen, die von **pcnfsd** bedient werden, fallen in drei verschiedene Kategorien: Authentisierung, Drucken und Diverses. Für die Systemverwaltung sind nur Authentisierung und Drucken interessant.

→

pcnfsd
←

Authentisierung

Wenn der **pcnfsd** eine PCNFSD_AUTH oder eine PCNFSD2_AUTH-Anfrage empfängt, wird der Benutzer am System angemeldet, sofern der Benutzername bekannt und das Paßwort korrekt ist. Der Dämon gibt dann die Benutzer- und Gruppennummer, das Heimatverzeichnis und die Maske für die Zugriffsrechte zukünftig angelegter Dateien zurück. Außerdem fügt **pcnfsd** der *wtmp*-Datenbank einen entsprechenden Eintrag an. Wenn diese Protokollführung nicht gewünscht ist, kann sie durch Eintragen der Zeile

```
wtmp off
```

in der */etc/pcnfsd.conf*-Datei abgeschaltet werden.

Drucken

pcnfsd unterstützt ein Druckmodell, das auf der Verwendung von NFS zum Transport der eigentlichen Druckdaten vom Client zum Server basiert. Der Client schickt zunächst eine PCNFSD_PR_INIT- oder PCNFSD2_PR_INIT-Anfrage, die der Server mit dem Pfad zum Spool-Verzeichnis beantwortet. Dieses Spool-Verzeichnis wird via NFS exportiert und muß vom Client zur Ablage der Druckdateien verwendet werden. **pcnfsd** legt für jeden Client ein eigenes Verzeichnis an, das normalerweise unterhalb von */usr/spool/pcnfs* liegt. Der Name des Verzeichnisses ist der Rechnername des Client-Rechners. Wenn das Spool-Verzeichnis an einer anderen Stelle angelegt werden soll, können Sie Ihrer Datei */etc/pcnfs.conf* die Zeile

spooldir *Pfad*

hinzufügen. Nachdem der Client das Spool-Verzeichnis bei sich eingehängt und die Druckerdaten in eine Datei in diesem Verzeichnis übertragen hat, führt **pcnfsd** eine PCNFSD_PR_START- oder PCNFSD2_PR_START-Anfrage aus. **pcnfsd** erzeugt basierend auf den Druckerdiensten des Server-Betriebssystems einen Druckbefehl und führt diesen unter dem Namen des PC-Benutzers aus. Jeder Druckjob enthält den Namen des zu verwendenden Druckers. **pcnfsd** interpretiert Drucker entweder als ein Druckziel, das vom Drucksubsystem bedient wird, oder als virtuelle Drucker. Virtuelle Drucker werden durch folgende Zeile in der */etc/pcnfsd.conf*-Datei definiert:

printer *Name Alias Befehl*

wobei *Name* der Name des zu definierenden virtuellen Druckers, *Alias* der tatsächliche Name eines existierenden Druckers und *Befehl* ein Befehl ist, der ausgeführt wird, wann immer eine Datei auf *Name* gedruckt wird.

pop2d

in.pop2d

Erlaubt es Benutzern, sich mit Port 109 zu verbinden und den Inhalt ihrer Mailbox in */var/spool/mail* abzurufen. **pop2d** erwartet den

Benutzernamen und ein Paßwort, bevor Nachrichten übersendet werden. Siehe auch **pop3d**.

pop2d

Befehle

Jeder Befehl muß auf einer separaten Zeile stehen.

HELO Fragt nach Benutzername und Paßwort.

FOLD Öffnet */var/spool/mail/$USER.*

HOST Öffnet */var/spool/pop/$USER.*

READ Liest eine Nachricht.

RETR Überträgt eine Nachricht.

ACKS Speichert die zuletzt übertragene Nachricht und springt zur nächsten.

ACKD Löscht die zuletzt übertragene Nachricht und springt zur nächsten.

NACK Speichert die zuletzt übertragene Nachricht und fordert sie erneut an.

QUIT Beenden.

in.pop3d

pop3d

pop3d ist eine neuere Version von **pop2d**. Das Verhalten ist ähnlich, die verwendeten Befehle sind aber geringfügig anders.

Befehle

USER Fragt nach dem Benutzernamen.

PASS Fragt nach dem Paßwort.

STAT Zeigt die Anzahl der Nachrichten in der Mailbox und ihre Gesamtgröße an.

LIST Zeigt die Größe der einzelnen Nachrichten an.

DELE Löscht eine Nachricht.

NOOP Keine Funktion.

LAST Gibt die Nummer der zuletzt übertragenen und gelesenen Nachricht aus.

RSET Setzt alle Löschmarkierungen zurück.

TOP Gibt den ersten Teil der Nachricht aus.

QUIT Beenden.

rpc.portmap [*Option*]

portmap

NFS/NIS-Befehl. Bildet RPC-Programmnummern auf IP-Ports ab. **portmap** ist ein Server, der RPC-Programmnummern in die zugehörigen IP-Portnummern konvertiert. Er muß gestartet sein, damit RPC-Aufrufe ausgeführt werden können. Wenn ein RPC-Server startet, teilt er **portmap** mit, an welchem Port er lauscht und welche RPC-Programmnummern er bedienen kann. Wenn ein Client einen RPC-Aufruf ausführen möchte, kontaktiert er zunächst **portmap** auf dem Server, um die Portnummer zu bestimmen, an welche die RPC-Pakete geschickt werden sollen. **portmap** muß der erste gestartete RPC-Server sein.

→

portmap
←

Option

-d Startet **portmap** im Debugging-Modus und nicht als Dämon.

ping

ping [*Optionen*] *Rechner*

Sicherstellen, daß ein entfernter Rechner am Netz ist und antwortet. **ping** ist zum Testen, Ausmessen und Verwalten von Netzwerken gedacht. Wegen der mit **ping** verbundenen Netzwerklast ist es nicht sinnvoll, **ping** im normalen Betrieb oder aus automatisierten Skripten heraus aufzurufen.

Optionen

-c *Anzahl*
: Nur *Anzahl* ECHO_RESPONSE-Pakete senden (und empfangen).

-d Schaltet die SO_DEBUG-Option beim verwendeten Socket ein.

-f Sendet ausgehende **ping**-Pakete in der gleichen Geschwindigkeit, in der Antwortpakete eingehen, mindestens aber 100 Pakete pro Sekunde. Dies ist eine schwere Belastung für ein Netzwerk und sollte mit Bedacht verwendet werden; nur der Superuser kann diese Option einsetzen.

-i *Sekunden*
: Wartet *Sekunden* Sekunden, bevor das nächste Paket gesendet wird. Der Default ist eine Sekunde. Diese Option ist inkompatibel mit der **-f**-Option.

-l *Anzahl*
: Schickt zunächst *Anzahl* Pakete so schnell wie möglich, bevor zum normalen Verhalten übergegangen wird.

-n Nur numerische Ausgabe; es wird nicht versucht, symbolische Namen für Rechner-Adressen nachzuschlagen.

-p *Bits*
: Schickt bis zu 16 Füllbytes in jedem Paket. Diese Option ist nützlich, um datenabhängige Netzwerkprobleme zu diagnostizieren. *Bits* wird in Hexadezimalzahlen angegeben. Beispielsweise schickt **-p ff** Pakete, die mit lauter Einsen gefüllt sind.

-q Stilles Arbeiten — nichts außer den zusammenfassenden Zeilen am Anfang und Ende wird angezeigt.

-r Umgeht die normalen Routing-Tabellen und schickt die Pakete direkt an einen Rechner eines angeschlossenen Netzwerks.

-s *Paketgröße*
: Gibt an, wie viele Datenbytes geschickt werden sollen. Der Default ist 56, was wegen der acht Bytes des ICMP-Headers zu 64 ICMP-Datenbytes führt.

-v Verbose — auch andere empfangene Pakete außer ICMP-Paketen werden angezeigt.

-R Einstellen der IP-Routenprotokolloption, mit der die Route eines Pakets im IP-Header verzeichnet wird. Der Inhalt der Route wird ausgegeben, wenn die Option **-v** angegeben wird,

und wird bei empfangenen Paketen eingetragen, wenn der angesprochene Rechner die Routen wieder zurückgeschickt hat oder die -l-Option eingeschaltet ist.

ping

powerd *Gerät*

powerd

Überwachen der Verbindung zu einer unterbrechungsfreien Stromversorgung, die der Benutzer via *Gerät* angibt. Wenn die Stromversorgung gestört wird, wird **init** von **powerd** veranlaßt, die **powerwait**- und **powerfail**-Einträge auszuführen. Wenn die Stromversorgung wiederhergestellt ist, wird **init** dazu veranlaßt, die **powerokwait**-Einträge auszuführen.

pppd [*Optionen*] [*Terminal*] [*Geschwindigkeit*]

pppd

PPP steht für Point-to-Point Protocol (Punkt-zu-Punkt-Protokoll). Mit PPP können Datagramme über eine serielle Verbindung übertragen werden. **pppd** versucht entweder das angegebene *Terminal*, das in */dev* vorhanden sein muß, oder das kontrollierende Terminal für PPP (Default) zu konfigurieren. Außerdem können Sie die Baudrate mit *Geschwindigkeit* einstellen.

Optionen

asyncmap *Tabelle*
: Angeben, welche Steuerzeichen nicht übertragen werden sollen. *Tabelle* sollte eine 32 Bit große Hexadezimalzahl sein, in der jedes Bit ein zu maskierendes Zeichen repräsentiert. Beispielsweise repräsentiert 00000001 das Zeichen 0x00 und 80000000 das Zeichen 0x1f oder _. Es können mehrere Zeichen angegeben werden.

auth
: Verlangt eine Selbst-Authentisierung der Gegenstelle, bevor die ersten Pakete übertragen werden.

connect *Befehl*
: Verbindet, wie in *Befehl* angegeben. *Befehl* kann eine ausführbare Binärdatei oder ein Shell-Befehl sein.

debug, -d
: Erhöht die Debugging-Ebene.

defaultroute
: Fügt eine neue Defaultroute hinzu, für welche die Gegenstelle als Gateway fungiert. Wenn die Verbindung beendet wird, wird die Route entfernt.

-detach
: Arbeitet im Vordergrund. Normalerweise erzeugt **pppd** eine Kopie von sich und arbeitet im Hintergrund.

disconnect *Befehl*
: Beendet die Verbindung, wie in *Befehl* angegeben. *Befehl* kann eine ausführbare Binärdatei oder ein Shell-Befehl sein.

SysAdmin Befehle

→

domain *d*
: Legt den Domänennamen *d* fest.

escape *Zeichenliste*
: Maskieren aller Zeichen in *Zeichenliste*, wobei es sich um eine durch Kommata getrennte Liste von Hexadezimalzahlen handeln soll. 0x20–0x3f und 0x5e können nicht maskiert werden.

file *Datei*
: Optionen aus *Datei* einlesen.

lock Nur **pppd** darf auf das verwendete Gerät zugreifen.

mru *Bytes*
: Pakete mit mehr als *Bytes* Bytes zurückweisen.

name *Name*
: Gibt den Rechnernamen für das lokale System an.

netmask *Maske*
: Angeben der Netzmaske (z.B. 255.255.255.0).

passive, -p
: Programm nicht beenden, wenn die Gegenstelle nicht auf den Wunsch zum Verbindungsaufbau antwortet. Statt dessen wird auf ein gültiges Paket von der Gegenstelle gewartet.

silent Keine Pakete schicken, bevor nicht eines empfangen wurde.

[*lokale_IP-Adresse*]:[*entfernte_IP-Adresse*]
: Angeben der lokalen und entfernten IP-Adresse, entweder als Rechnernamen oder als numerische Adressen.

Dateien

/var/run/pppn.pid
: Prozeßnummer von **pppd**.

/etc/ppp/ip-up
: Binärdatei oder Skript, die oder das ausgeführt wird, wenn die PPP-Verbindung aktiv wird.

/etc/ppp/ip-down
: Binärdatei oder Skript, die oder das ausgeführt wird, wenn die PPP-Verbindung beendet wird.

/etc/ppp/pap-secrets
: Benutzernamen, Paßwörter und IP-Adressen für die Authentisierung via PAP.

/etc/ppp/chap-secrets
: Benutzernamen, Paßwörter und IP-Adressen für die Authentisierung via CHAP.

/etc/ppp/options
: Systemdefaults. Die Optionen in dieser Datei werden *vor* denen der Kommandozeile gesetzt.

~/.ppprc
: Die Defaultoptionen des Benutzers. Diese werden vor den Kommandozeilenoptionen, aber nach den Systemdefaults gelesen.

/etc/ppp/options.ttyname
Der Name der defaultmäßigen seriellen Schnittstelle.

pppd

psupdate [*Tabellendatei*]

psupdate

Aktualisieren der **psupdate**-Datenbank (auf manchen Systemen */boot/psupdate*, auf anderen */etc/psdatabase*), die Informationen über die Tabellendatei (*map file*) des Kernelimages enthält. Wenn keine Tabellendatei angegeben wird, wird die Default-Datei verwendet (je nach Distribution */usr/src/linux/vmlinux* oder */usr/src/linux/tools/zSystem*).

ramsize [*Option*] [*Image* [*Größe* [*Position*]]]

ramsize

Wenn keine Optionen angegeben werden, werden Benutzungs-Informationen zur RAM-Disk ausgegeben. Das Bytepaar an Position 504 im Kernelimage gibt normalerweise die Größe der RAM-Disk an. Mit *Image* kann ein Kernelimage angegeben werden, mit *Position* eine andere Position im Image. Wenn *Größe* angegeben wird, wird eine neue Größe für die RAM-Disk angegeben. **rdev -r** ist identisch mit **ramsize**.

Option

-o *Position*
Wie *Position* als Argument.

rarp [*Optionen*]

rarp

Verwalten der Tabelle des Reverse Address Resolution Protocols (RARP). Die Tabelle ist normalerweise die virtuelle Datei */proc/net/rarp*.

Optionen

-a [*Rechnername*]
Anzeigen aller Einträge. Wenn *Rechnername* angegeben wird (darf eine Liste sein), werden nur die darauf passenden Einträge angezeigt.

-d *Rechnername*
Entfernt die auf *Rechnername* (darf eine Liste sein) passenden Einträge.

-s *Rechnername HW-Adr*
Fügt einen neuen Eintrag für *Rechnername* mit der Hardware-Adresse *HW-Adr* hinzu.

-t *Typ*
Berücksichtigt beim Auslesen oder Ändern der Tabelle nur Einträge des Typs *Typ*. *Typ* kann entweder **ether** (Default) oder **ax25** sein.

-v
Verbose-Modus.

rdate [*Option*]

rdate

TCP/IP-Befehl. Teilt dem Time Server mit, daß sich Datum oder Uhrzeit geändert hat. Wenn der lokale Time Server ein Master-Server ist, teilt er allen Clients mit, daß sich Datum oder Uhrzeit verändert hat. Wenn der

→

SysAdmin Befehle

rdate ←

lokale Time Server ein Slave-Server ist, bittet er den Master-Server, Datum oder Uhrzeit zu aktualisieren.

Option

-p Datum ausgeben, aber nicht aktualisieren.

rdev

rdev [*Optionen*] [*Image* [*Wert* [*Position*]]]

Wenn keine Argumente angegeben werden, wird eine Zeile in */etc/mtab*-Syntax angezeigt, die das Wurzel-Dateisystem beschreibt. Anderenfalls ändert **rdev** die Werte der Bytes im Kernelimage, welche die Größe der RAM-Disk (defaultmäßig Position 504), den VGA-Modus (defaultmäßig Position 506) und das Gerät, welches das Wurzelverzeichnis enthält (defaultmäßig Position 508), angeben. Das zu ändernde Kernel*image* muß, der neue *Wert* oder eine andere *Position* kann angegeben werden.

Optionen

-o *Position*
: Wie *Position* als Argument.

-r **rdev** verhält sich wie **ramsize**.

-s **rdev** verhält sich wie **swapdev**.

-v **rdev** verhält sich wie **vidmode**.

-R **rdev** verhält sich wie **rootflags**.

rdist

rdist [*Optionen*] [*Namen*]

Client-Programm für die Verteilung von Programmen auf andere Rechner. **rdist** hat identische Kopien von Dateien auf mehreren Rechnern. Das Programm liest Befehle aus einer Distributionsdatei *distfile* ein, die das Aktualisieren der Dateien oder Verzeichnisse steuern. Die Distributionsdatei wird mit der **-f**, der **-c** oder **-** angegeben; im letzteren Fall wird sie von der Standardeingabe eingelesen.

Optionen

-a *Anzahl*
: Keine Dateisysteme aktualisieren, auf denen weniger als *Anzahl* Bytes frei sind.

-c *Name* [*Benutzername*@]*Rechner*[:*Ziel*]
: ([*Benutzername*@]*Rechner*[:*Ziel*]) wird als kleine *Distributionsdatei* interpretiert

-d *Var*=*Wert*
: Definiert die Variable *Var* mit dem Wert *Wert*. Mit dieser Option werden Variablen neu definiert oder Variablendefinitionen in der *Distributionsdatei* überschrieben.

-f *Datei*
: Eingabe aus *Datei* lesen (der Default ist *distfile*). Wenn *Datei* - ist, wird von der Standardeingabe gelesen.

-l *Optionen*
Optionen zum Protokollieren auf dem lokalen Rechner angeben.

-m *Rechner*
Nur *Rechner* aktualisieren. Kann mehrfach für mehrere *Rechner* angegeben werden.

-n
Normale Ausführung unterdrücken und statt dessen die Befehle anzeigen, die ausgeführt worden wären.

-o*Optionen*
Eine oder mehrere durch Kommata getrennte Optionen angeben:

chknfs Keine Dateien bearbeiten, die auf NFS-Dateisystemen liegen.

chkreadonly
Dateisystem überprüfen, ob es auch schreibbar ist, bevor eine Aktualisierung versucht wird.

chksym Keine Dateien aktualisieren, die auf dem lokalen Rechner liegen, aber symbolische Links auf Dateien auf einem entfernten Rechner sind.

compare Dateien vergleichen; dieser Vergleich wird als Kriterium dafür herangezogen, welche Dateien aktualisiert werden müssen (normalerweise das Datum der letzten Änderung).

follow Symbolische Links interpretieren; Kopieren der Datei, auf die ein Link zeigt, anstatt auf der entfernten Maschine einen symbolischen Link anzulegen.

ignlnks Links, die unauflösbar erscheinen, ignorieren.

nochkgroup
Nicht die Gruppen-ID einer Datei aktualisieren, es sei denn, die ganze Datei muß aktualisiert werden.

nochkmode
Nicht die Zugriffsrechte einer Datei aktualisieren, es sei denn, die ganze Datei muß aktualisiert werden.

nochkowner
Nicht die Benutzer-ID einer Datei aktualisieren, es sei denn, die ganze Datei muß aktualisiert werden.

nodescend
Nicht rekursiv in Verzeichnisse absteigen.

noexec Keine ausführbaren Dateien, die im *a.out*-Format sind, verteilen.

numchkgroup
Gruppenrechte werden anhand der ID anstelle des Namens überprüft.

numchkowner
Der Besitzer wird anhand der ID anstelle des Namens überprüft.

→

rdist ←

quiet	Stiller Modus; ausgeführte Befehle werden nicht ausgegeben.
remove	Dateien, die auf dem entfernten Rechner, aber nicht auf dem lokalen vorhanden sind, entfernen.
savetargets	Aktualisierte Dateien in *Name.old* sichern.
verify	Eine Liste aller Dateien auf dem entfernten Rechner, die aktualisiert werden müßten, ausgeben, aber die Aktualisierung selbst nicht vornehmen.
whole	Erhalten von Verzeichnisstrukturen durch Anlegen von Unterverzeichnissen auf dem entfernten Rechner. Wenn beispielsweise die Datei */foo/bar* in das Verzeichnis */baz* verteilt wird, wird daraus */baz/foo/bar* anstelle des Defaults */baz/bar*.
younger	Keine Dateien aktualisieren, die jünger als die Master-Dateien sind.

-p *Pfad*
: Den Pfad angeben, in dem **rdistd** auf dem entfernten Rechner gesucht wird.

-t *Sekunden*
: Angeben der Timeout-Periode (Default: 900 Sekunden), nach der **rdist** die Verbindung zu einem entfernten Server schließt, wenn dieser noch nicht geantwortet hat.

-A *Num*
: Minimale Anzahl an Inodes, die **rdist** erwartet, angeben.

-D
: Debugging-Modus.

-F
: Alle Befehle sequentiell, d.h. ohne Starten neuer Prozesse, ausführen.

-L *Optionen*
: Logging-Optionen auf dem entfernten Rechner angeben.

-M *Anzahl*
: Nicht mehr als *Anzahl* Kindprozesse von **rdist** simultan ablaufen lassen. Der Default ist 4.

-P *Pfad*
: Den Pfad zu *rsh* auf der lokalen Maschine angeben.

---

rdistd

rdistd *Optionen*

Starten des **rdist**-Servers. Es *muß* grundsätzlich die -S-Option angegeben werden, es sei denn, Sie erfragen nur mit -V die Versionsinformation.

Optionen

-D
: Debugging-Modus.

-S
: Server starten.

-V
: Versionsnummer ausgeben und Programm beenden.

reboot [*Optionen*]

reboot

Dateisysteme schließen, System herunterfahren und neu starten. Da dieser Befehl alle Prozesse sofort anhält, sollte er nur im Single-User-Modus (Wartungsmodus) verwendet werden. Wenn sich das System nicht in Runlevel 0 oder 6 befindet, ruft **reboot** statt dessen **shutdown -nf** auf.

Optionen

- -d Nicht nach */var/log/wtmp* schreiben.
- -f Auch dann **reboot** aufrufen, wenn normalerweise **shutdown** aufgerufen werden würde.
- -n Den normalen Aufruf von **sync** unterdrücken.
- -w Normale Ausführung unterdrücken; nur nach */var/log/wtmp* schreiben.

rexecd *Befehlszeile*

rexecd

TCP/IP-Befehl. Server für die **rexec**-Routine. **rexecd** ermöglicht es, Programme auf entfernten Rechnern auszuführen. Die Authentisierung geschieht dabei über Benutzernamen und Paßwörter. **rexecd** wird von **inetd** gestartet und muß daher in der Konfigurationsdatei von **inetd**, */etc/inetd.conf*, eingetragen sein. Wenn **rexecd** eine Service-Anfrage erhält, läuft das folgende Protokoll ab:

1. Der Server liest alle Zeichen aus dem Socket, bis ein Nullbyte auftritt. Der resultierende String wird als ASCII-Dezimalzahl interpretiert.
2. Wenn die in Schritt 1 erhaltene Zahl nicht Null ist, wird sie als Portnummer eines zweiten Streams interpretiert, der für **stderr** verwendet wird. Es wird eine zweite Verbindung zu dem angegebenen Port auf dem Client-Rechner aufgebaut.
3. Ein höchstens sechzehn Zeichen langer und mit einem Nullbyte terminierter Benutzername wird vom ursprünglichen Socket gelesen.
4. Ein höchstens sechzehn Zeichen langes und mit einem Nullbyte terminiertes, nicht verschlüsseltes Paßwort wird vom ursprünglichen Socket gelesen.
5. Ein mit einem Nullbyte terminierter Befehl wird vom ursprünglichen Socket gelesen. Die Länge des Befehls ist durch die maximale Größe der Argumentliste auf dem System beschränkt.
6. **rexecd** überprüft dann Benutzername sowie Paßwort und, falls die Authentisierung erfolgreich verlief, wechselt in das Heimatverzeichnis und richtet die Benutzer- und Gruppenzugriffsrechte ein.
7. Über die mit **stderr** geknüpfte Verbindung wird ein Nullbyte geschickt. Die bereits geholte Kommandozeile wird an die normale Anmeldeshell des Benutzers weitergegeben. Die Shell erbt die von **rexecd** aufgebauten Verbindungen.

→

rexecd
←

Fehlermeldungen

`username too long`
Der Benutzername ist länger als sechzehn Zeichen.

`password too long`
Das Paßwort ist länger als sechzehn Zeichen.

`command too long`
Der übergebene Befehl ist größer als erlaubt.

`Login incorrect`
Der angegebene Benutzer ist dem System nicht bekannt.

`Password incorrect`
Es wurde ein falsches Paßwort angegeben.

`No remote directory`
Es konnte nicht in das Heimatverzeichnis verzweigt werden.

`Try again`
Der Server konnte kein Duplikat seines Prozesses anlegen.

`<shellname>: . . .`
Der Server konnte kein Duplikat seines Prozesses anlegen. Die Anmeldeshell des Benutzers konnte nicht gestartet werden.

---

rlogind

rlogind [*Optionen*]

TCP/IP-Befehl. Der Server für das **rlogin**-Programm. Er ermöglicht Benutzern entfernter Rechner das Anmelden auf diesem Rechner. Die Authentisierung basiert auf privilegierten Portnummern auf vertrauenswürdigen Rechnern. **rlogind** wird von **inetd** aufgerufen, wenn ein entferntes Anmelden durchgeführt werden soll, und führt das folgende Protokoll aus:

- Der Server überprüft den Port des Clients, von dem die Verbindung ausgeht. Wenn die Portnummer nicht im Bereich 0-1023 ist, bricht der Server die Verbindung ab.
- Der Server überprüft die Adresse des Clients und versucht, den dazugehörigen Rechnernamen herauszufinden. Wenn der Rechnername nicht bestimmt werden kann, wird die Punkt-Notation der Rechneradresse verwendet.

Im Zuge des Anmeldevorgangs werden die Baudrate und der Terminaltyp des Terminals des Clients, wie er in der Umgebungsvariablen TERM steht, übermittelt.

Optionen

-a Rechnernamen überprüfen.

-l Keine Authentisierung von Rechnern über eine *.rhosts*-Datei, die nicht **root** gehört, vornehmen.

-n Keep-alive Nachrichten unterdrücken.

rmail *user*...

rmail

TCP/IP-Befehl. Bearbeitet einkommende Email von anderen Rechnern, die via **uucp** empfangen wurde. Dabei werden die `From`-Zeilen in der von **mail** generierten Form in eine einzige Zeile der Form `Rückleitweg!Absender` umgewandelt. Die so verarbeitete Nachricht wird an **sendmail** zur weiteren Zustellung weitergeleitet. **rmail** ist speziell für die Zusammenarbeit mit **uucp** und **sendmail** entwickelt worden.

rmmod [*Option*] *Module*

rmmod

Ein Modul oder eine Liste von Modulen aus dem Kernel entladen. Dies kann nur erfolgreich durchgeführt werden, wenn die angegebenen Module weder in Gebrauch sind noch von anderen Modulen verwendet werden.

Option

-r
: Module rekursiv entfernen (alle Module, die das angegebene Modul benutzen).

rootflags [*Option*] *Image* [*Schalter* [*Position*]]

rootflags

Setzen der *Schalter* in einem Kernelimage. Wenn keine Argumente angegeben werden, werden die *Schalter* des Kernel*images* ausgegeben. Bei *Schalter* handelt es sich um eine zwei Byte große Ganzzahl, die an Position 498 im Kernel*image* steht. Zur Zeit ist der einzige Effekt dieses Schalters, daß das Wurzeldateisystem nur zum Lesen eingehängt wird, wenn der Wert von *Schalter* ungleich null ist. Sie können die *Schalter* verändern, indem Sie das zu ändernde Kernel*image* und die neuen *Schalter* angeben. Außerdem kann auch eine andere Position angegeben werden (der Default ist 498). **rdev -R** ist ein Synonym für **rootflags**. Wenn **LILO** verwendet wird, wird **rootflags** nicht gebraucht. Die *Schalter* können dann auch beim Booten von **LILO** aus gesetzt werden.

Option

-o *Position*
: Wie *Position* als Argument.

SysAdmin Befehle

route [*Option*] [*Befehl*]

route

TCP/IP-Befehl. Manipulieren der Routing-Tabellen, die normalerweise von **routed** verwaltet werden. **route** kennt zwei Befehle: **add**, um eine Route hinzuzufügen, und **del**, um eine Route zu löschen. Die beiden Befehle haben die folgende Syntax:

add [-net | -host] *Adresse* **[gw** *Gateway***]**
[netmask *Maske***] [mss** *TCP-MSS***] [dev** *Gerät***]**
del *Adresse*

Adresse wird als normale Route behandelt, wenn nicht **-net** angegeben wird oder *Adresse* in */etc/networks* steht. **-host** kann verwendet werden, um anzuzeigen, daß es sich bei *Adresse* um eine normale Route handelt,

→

route ←

unabhängig davon, ob die Adresse in */etc/networks* steht. Das Schlüsselwort **default** gibt an, daß diese Route immer dann zu verwenden ist, wenn keine andere bekannt ist. Außerdem können Sie angeben: einen *Gateway*, durch den Pakete für die *Adresse* geroutet werden, eine Netzmaske *Maske* und die maximale Segmentgröße des TCP-Protokolls *TCP-MSS* sowie das *Gerät*, das mit der Route verbunden ist. Die Routing-Tabellen können nur vom Superuser verändert werden.

Wenn kein Befehl angegeben wird, gibt **route** die Routing-Tabellen aus.

Option

- -n Rechner- und Netzwerknamen nicht symbolisch ausgeben, wenn Aktionen protokolliert werden.

routed

routed [*Optionen*] [*Logdatei*]

TCP/IP-Befehl. Netzwerk-Routing-Dämon. **routed** wird vom Superuser während des Hochfahrens des Systems gestartet, um die Internet-Routing-Tabellen zu verwalten. Der Dämon verwendet dabei eine Variante des Xerox NS Routing Information Protocols, um die Einträge in der Routing-Tabelle des Kernels aktuell zu halten. Nach dem Start verwendet **routed** den SIOCGIFCONF-ioctl-Aufruf, um diejenigen Netzwerkschnittstellen zu finden, die direkt mit dem System verbunden und aktiv sind. **routed** schickt ein REQUEST-Paket an jede Schnittstelle und läuft dann in einer Schleife, in der er auf REQUEST- und RESPONSE-Pakete von anderen Rechnern wartet. Wenn ein REQUEST- Paket eintrifft, stellt **routed** basierend auf der in seinen internen Tabellen enthaltenen Information eine Antwort zusammen. Das so generierte RESPONSE-Paket enthält eine Liste der bekannten Routen. Alle empfangenen RESPONSE-Pakete werden dazu verwendet, die Routing-Tabellen entsprechend zu aktualisieren.

Bei jeder Aktualisierung verzeichnet **routed** die Änderung in seinen internen Tabellen, aktualisiert die Routing-Tabelle des Kernels und erzeugt ein RESPONSE-Paket, das diese Änderungen wiedergibt und an alle direkt angeschlossenen Rechner und Netzwerke übertragen wird.

Optionen

- -d Debugging-Modus. Zusätzliche Informationen in *Logdatei* mitprotokollieren.
- -g Eine Route zum Defaultziel anbieten.
- -q Das Gegenteil von -s.
- -s Erzwingt, daß **routed** Routing-Information herausgibt, unabhängig davon, ob er als Router zwischen Netzwerken fungiert oder nicht.
- -t Verhindert, daß **routed** sich in den Hintergrund legt und sich vom kontrollierenden Terminal abkoppelt. Damit beenden Interrupts von der Tastatur den Prozeß.

rpcinfo [*Optionen*] [*Rechner*] [*Programm*] [*Version*]

NFS/NIS-Befehl. Ausgeben von RPC-Informationen. *Programm* kann entweder ein Name oder eine Nummer sein. Wenn *Version* mit angegeben wird, versucht **rpcinfo**, die entsprechende Version des angegebenen *Programms* aufzurufen. Ansonsten versucht es, alle registrierten Versionen von *Programm* zu finden, indem Version 0 aufgerufen und versucht wird, jede registrierte Version aufzurufen.

Optionen

-b *Programm Version*
: Ausführen eines RPC-Broadcasts via UDP an das angegebene *Programm* mit der angegebenen *Version* und Ausgeben der Namen aller Rechner, die darauf antworten.

-d *Programm Version*
: Löschen der *Version* von *Programm* aus der Registrierungsdatenbank. Kann nur vom Superuser oder von demjenigen Benutzer, dessen Prozeß die Registrierung vorgenommen hat, ausgeführt werden.

-n *Portnummer*
: Bei Verwendung von -t und -u den Port *Portnummer* verwenden.

-p [*Rechner*]
: Ausgeben einer Liste aller registrierten RPC-Programme auf *Rechner*. Wenn *Rechner* weggelassen wird, wird die Ausgabe des **hostname**-Befehls verwendet.

-t *Rechner Programm* [*Version*]
: Durchführen eines RPC-Aufrufs via TCP an *Programm* auf dem *Rechner* und ausgeben, ob eine Antwort empfangen wurde.

-u *Rechner Programm* [*Version*]
: Durchführen eines RPC-Aufrufs via UDP an *Programm* auf dem *Rechner* und ausgeben, ob eine Antwort empfangen wurde.

Beispiele

Anzeigen aller RPC-Dienste, die auf der lokalen Maschine registriert sind:

```
$ rpcinfo -p
```

Anzeigen aller RPC-Dienste, die auf dem Rechner **ratbert** installiert sind:

```
$ rpcinfo -p ratbert
```

Anzeigen aller Maschinen im lokalen Netz, auf denen der Network Information Service (NIS) läuft:

```
$ rpcinfo -b ypserv Version | uniq
```

wobei *Version* die aktuelle NIS-Versionsnummer ist, die Sie aus dem oben angegebenen -p-Schalter erhalten.

rshd	**rshd** [*Optionen*] TCP/IP-Befehl. Server für das Starten von nicht-interaktiven Shells auf entfernten Rechnern, also für Befehle wie **rcmd** oder **rcp**. **rshd** wird von **inetd** gestartet und muß in der Konfigurationsdatei von **inetd**, */etc/inetd.conf*, eingetragen sein. Die Optionen entsprechen genau denen von **rlogind**, mit Ausnahme der Option -L, die es nur bei **rshd** gibt. *Option* -L Alle erfolgreichen Verbindungen und fehlgeschlagenen Versuche via **syslogd** protokollieren.
rstat	**rstat** *Rechner* TCP/IP-Befehl. Faßt den Zustand von *Rechner* zusammen: die aktuelle Uhrzeit, die Zeit seit dem Systemstart, die durchschnittliche Last und die durchschnittliche Anzahl von Jobs, die auf ihre Ausführung warten. **rstat** fragt den **rstat_svc**-Dämon des entfernten Rechners ab.
run-parts	**run-parts** [*Optionen*] [*Verzeichnis*] Alle Skripten in *Verzeichnis* in alphabetischer Reihenfolge ausführen. Skripten, deren Dateinamen nicht-alphanumerische Zeichen (mit Ausnahme von Unterstrichen und Bindestrichen) enthalten, werden nicht ausgeführt. *Optionen* -- Alle folgenden Argumente als Dateinamen und nicht als Optionen interpretieren. --test Ausgeben, welche Skripten ausgeführt werden würden, aber das eigentliche Ausführen unterdrücken. --umask=*umask* Angeben der *umask*, der Maske für die Zugriffsrechte zukünftig angelegter Dateien. Der Default ist 022.
runlevel	**runlevel** Ausgeben der vorhergehenden und aktuellen Runlevels des Systems.
ruptime	**ruptime** [*Optionen*] TCP/IP-Befehl. Anzeigen, wie lange jeder Rechner im lokalen Netzwerk seit seinem letzten Systemstart bereits läuft und wie viele Benutzer auf ihm angemeldet sind. Wenn ein Rechner sich elf Minuten lang nicht gemeldet hat, wird angenommen, daß er nicht in Betrieb ist.

Optionen

- -a Auch Benutzer ausgeben, die mehr als eine Stunde keinen Befehl mehr eingegeben haben.
- -l Rechner nach durchschnittlicher Last sortieren.
- -r Normale Sortierreihenfolge umdrehen.
- -t Maschinen nach Laufzeiten seit dem letzten Systemstart sortieren.
- -u Rechner nach der Anzahl der angemeldeten Benutzer sortieren.

ruptime

rusers [*Optionen*] [*Rechner*]

rusers

TCP/IP-Befehl. Ausgeben, welche Benutzer auf *Rechner* oder auf allen lokalen Rechnern angemeldet sind. Die Ausgabe geschieht im Format des **who**-Befehls.

Optionen

- -a Auch Maschinen ausgeben, auf denen niemand angemeldet ist.
- -l Weitere Informationen ausgeben: Terminal, Datum, Uhrzeit, Zeit ohne Befehlseingabe, entfernter Rechner.

rwall *Rechner* [*Datei*]

rwall

TCP/IP-Befehl. Gibt eine Nachricht an alle Benutzer aus, die auf *Rechner* angemeldet sind. Wenn *Datei* angegeben ist, wird die Meldung daraus entnommen, ansonsten wird sie von der Standardeingabe gelesen.

rwhod

rwhod

TCP/IP-Befehl. Server für den Systemzustand, der die von **rwho** und **ruptime** verwendete Datenbank pflegt. Die Funktionalität basiert auf der Fähigkeit, Nachrichten an alle Rechner eines Netzwerks schicken zu können. Als Produzent von Informationen fragt **rwhod** periodisch den Zustand des Systems ab und erzeugt daraus Zustandsmeldungen, die an alle Rechner im Netzwerk geschickt werden. Als Konsument von Informationen lauscht **rwhod** den Zustandsmeldungen von **rwhod**-Servern auf anderen Systemen, überprüft diese und speichert sie dann in einer Reihe von Dateien im Verzeichnis */var/spool/rwho* ab. Nachrichten, die der **rwhod**-Server empfängt, werden verworfen, wenn sie nicht am Serverport von **rwhod** eintreffen. Zustandsmeldungen werden etwa alle drei Minuten erzeugt.

script [*Option*] [*Datei*]

script

Erzeugt eine Kopie der aktuellen Shell und protokolliert eine Sitzung am Terminal mit. Das Sitzungsprotokoll wird nach *Datei* geschrieben. Wenn keine *Datei* angegeben wird, wird das Protokoll in der Datei *typescript* im aktuellen Verzeichnis abgelegt. Die Protokollierung endet, wenn die Kopie der Shell beendet wird.

→

SysAdmin Befehle

script
←

Option

-a — Anhängen des Protokolls an den Inhalt von *Datei* oder *typescript*, anstatt den vorherigen Inhalt zu überschreiben.

sendmail

sendmail [*Schalter*] [*Adresse ...*]

sendmail ist ein Programm zum Weiterleiten von elektronischen Nachrichten (Email). Es nimmt Nachrichten von Email-Benutzerprogrammen entgegen, ersetzt die Anschrift durch eine Form, die vom zustellenden Programm verstanden wird, und leitet die Email an das Zustellungsprogramm weiter.

Kommandozeilenschalter

-b*x* — Einstellen des Betriebsmodus auf *x*. Die Betriebsmodi sind:
- a — ARPA-Modus.
- d — Als Dämon starten.
- i — Initialisieren der Alias-Datenbank.
- m — Nachrichten zustellen (Default).
- p — Ausgeben der Nachrichten-Warteschlange.
- s — Bei **sendmail** eingehende Nachrichten via SMTP entgegennehmen.
- t — Test-Modus.
- v — Nur Adressen überprüfen, aber keine Zustellung veranlassen.

-C*Datei*
: Verwenden der Konfigurationsdatei *Datei*.

-d*Ebene*
: Einstellen der Debugging-Ebene.

-F*Name*
: Einstellen des vollen Namens des Benutzers auf *Name*.

-f*Name*
: Der Name des Absenders ist *Name*.

-h*Anzahl*
: Den Hop-Zähler (zählt, wie oft eine Nachricht von **sendmail** verarbeitet wurde) auf *Anzahl* setzen.

-n — Keine Aliasnamen verwenden, nicht weiterleiten.

-o*x Wert*
: Option **x** auf den Wert *Wert* setzen. Die Optionen werden weiter unten beschrieben.

-p*Protokoll*
: Nachrichten mit dem angegebenen *Protokoll* empfangen.

-q*Zeit* — In der Warteschlange stehende Nachrichten verarbeiten. Wenn *Zeit* angegeben wird, wird die Warteschlange so oft geleert, wie *Zeit* angibt.

-r*Name*
: Veraltete Form von -f.

-t Auslesen der Header-Felder `To:`, `Cc:` und `Bcc:`. Die Mail wird an jeden Empfänger auf dieser Liste geschickt.

-v Verbose-Modus.

-X *Datei*
Sämtlichen Email-Verkehr in *Datei* mitprotokollieren. Sollte nicht für normales Logging verwendet werden.

Konfigurationsoptionen

Die folgenden Optionen können von der Kommandozeile mit dem -o-Schalter oder einer O-Zeile in der Konfigurationsdatei eingestellt werden:

7 Alle eingehenden Nachrichten auf sieben Bits formatieren.

a*Min* Wenn die Option D eingeschaltet ist, wird *Min* Minuten darauf gewartet, daß die Alias-Datenbank neu aufgebaut wird, bevor eine Warnung ausgegeben wird, daß die Alias-Datenbank veraltet ist.

A*Datei* Andere Alias-Datei verwenden.

b*MinBlocks/MaxGröße*
Verlangt, daß mindestens *MinBlocks* frei bleiben. Die maximale Größe einer Nachricht wird auf *MaxGröße* gesetzt. Wenn *MaxGröße* weggelassen wird, ist der Schrägstrich optional.

B*Zeichen*
Einstellen des Ersetzungszeichens für nicht gequotete Leerzeichen.

c Wenn eine Verbindung als „teuer" gilt, nicht sofort eine Verbindung aufbauen.

C*Anzahl*
Kontrollieren und Aktualisieren der Warteschlange, wenn eine Nachricht an mehrere Empfänger gesendet wird. **sendmail** schreibt die Liste der Empfänger neu, nachdem jeweils *Anzahl* Empfänger verarbeitet worden sind.

d*x* Einstellen des Zustellungsmodus. Die vorhandenen Zustellungsmodi sind **i** für interaktive (synchrone) Zustellung, **b** für Zustellung im Hintergrund (asynchron), **q** für Einreihen der Nachricht in die Warteschlange (Zustellung erst, wenn nächstes Mal die Warteschlange geleert wird) und **d** für Einreihen in die Warteschlange, ohne daß DNS-Anfragen vorgenommen werden.

D Alias-Datenbank, wenn nötig, automatisch neu aufbauen.

e*x* Einstellen des Fehlerverarbeitungsmodus. Gültige Modi sind **m** für Zurücksenden einer Fehlernachricht, **w** für Zurückschreiben einer Fehlernachricht, **p** für Ausgeben der Fehlermeldung auf das Terminal (Default), **q** für Verwerfen der Fehlermeldung und **e** für spezielle, im BerkNet verwendete Fehlerbehandlung.

→

SysAdmin Befehle

sendmail
←

E*Text* Einstellen des Headers für Fehlermeldungen. Bei *Text* handelt es sich entweder um Text, welcher der Fehlermeldung hinzugefügt werden soll, oder einen Dateinamen. Ein Dateiname muß mit vollem Pfad angegeben werden und mit einem Schrägstrich beginnen.

f UNIX-gemäße „`From:` "-Zeilen am Anfang einer Nachricht nicht entfernen.

F*Modus*
Default-Zugriffsrechte für temporäre Dateien einstellen. Wenn diese Option nicht angegeben wird, sind die Default-Zugriffsrechte 0644.

G Lokale Email-Namen mit dem GECOS-Feld in der Paßwort-Datei vergleichen.

g *n* Zu verwendende Gruppen-ID, wenn Mailprogramme aufgerufen werden.

H*Datei* SMTP-Hilfedatei.

h *Anzahl*
Maximal *Anzahl* Hops pro Nachricht erlauben.

i Einzelne Punkte auf einer Zeile nicht als Ende einer Nachricht interpretieren.

I *Arg* DNS-Anfragen verwenden und optimieren. Wenn keine DNS-Verbindungen zustande kommen, werden die Nachrichten in die Warteschlange eingereiht. Das *Arg*-Argument ist genauso wie die Schalter von **resolver**, nur ohne das RES_-Präfix. Jedem Schalter kann ein Plus- oder Minuszeichen vorangestellt werden, um die zugehörige Name-Server-Option ein- oder auszuschalten. Zwischen dem I und dem ersten Schalter muß ein Leerzeichen oder ein Tabulator stehen.

j MIME-Format für Fehlermeldungen verwenden.

J*Pfad* Ein alternativer Suchpfad für *.forward*.

k*Anzahl*
Größe des Verbindungscaches angeben.

K*Zeit* Timeout für Verbindungen auf *Zeit* einstellen.

l `Errors-To:`-Header nicht ignorieren.

L*n* Logging-Stufe angeben.

m Nachricht auch an den Absender schicken, wenn er auf einer Aliasliste steht.

M*XWert*
Definition eines Makros auf der Kommandozeile. Zuweisung von *Wert* an das Makro *X*.

n Während der Ausführung von **newaliases** die rechte Seite der Aliase validieren.

o Wenn diese Option eingeschaltet ist, dürfen die Nachrichten auch Header des alten Typs haben, ansonsten müssen sie neue Header haben (d.h. Kommata statt Leerzeichen zwischen den Empfängern).

p*was,was, . . .*
: Einstellen, wie privat oder öffentlich sich der SMTP-Dämon verhalten soll. Die einzelnen *was*-Argumente sollten voneinander durch Kommata getrennt werden. Die *was*-Argumente können unter den folgenden ausgewählt werden:

public SMTP voll öffentlich machen (Default).

needmailhelo Verlangt, daß ein anderer Rechner zunächst HELO oder EHLO sendet, bevor eine Nachricht geschickt wird.

needexpnhelo Verlangt, daß ein anderer Rechner zunächst HELO oder EHLO sendet, bevor eine Adreßexpansionsanfrage beantwortet wird.

needvrfyhelo Wie oben, aber für Verifizierungsanfragen.

noexpn Alle Expansionsanfragen ablehnen.

novrfy Alle Verifikationsanfragen ablehnen.

authwarnings Besondere Header in Nachrichten einfügen, die den Empfängern mitteilen, daß die Nachricht möglicherweise nicht authentisch ist.

goaway Alle oben stehenden Optionen außer **public** einschalten.

restrictmailq Benutzer derselben Gruppe wie die Besitzer des Warteschlangenverzeichnisses dürfen die Warteschlange einsehen.

restrictqrun Nur der Superuser und der Besitzer des Warteschlangenverzeichnisses dürfen das Abarbeiten der Warteschlange initiieren.

P*Benutzer*
: Kopien jeder nicht zustell- oder weiterleitbaren Nachricht an *Benutzer* (üblicherweise **postmaster**) schicken.

q*fact* Multiplikator für das Einstellen von Nachrichten in die Warteschlange bei hoher Systemlast.

Q*Schlangenverzeichnis*
: Auswahl des Verzeichnisses, in dem Nachrichten auf Zustellung warten.

R Keine Routen-Adressen entfernen.

S*Datei* Statistiken in *Datei* abspeichern.

s Immer eine Warteschlangendatei anlegen, auch dann, wenn es nicht unbedingt notwendig ist.

T*Zeit* Einstellen des Timeouts für nicht zugestellte Nachrichten in der Warteschlange auf die angegebene *Zeit*.

SysAdmin Befehle

→

t*Stz, Dtz*
Einstellen des Namens der Zeitzone.

U*Datenbank*
Benutzerdatenbank *Datenbank* für Weiterleitungsinformationen zu Rate ziehen.

u*N* Default-Benutzer-ID für Mailprogramme einstellen.

v Verbose-Modus.

V*Rechner*
Notfall-MX-Rechner. *Rechner* sollte der voll-qualifizierte Domänenname des Notfall-Rechners sein.

w Verwenden eines DNS-A-Datensatzes, wenn der MX-Datensatz nicht eindeutig ist.

x*Last* Nachrichten nicht weiterleiten, sondern in die Warteschlange stellen, wenn die Systemlast höher als *Last* ist.

X*Last* SMTP-Verbindungen zurückweisen, wenn die Systemlast höher als *Last* ist.

y*Faktor*
Lange Empfängerlisten mit dem *Faktor* bestrafen.

Y Jeden Job, der von der Warteschlange ausgeführt wird, in einem eigenen Prozeß starten. Dies trägt auf Systemen mit wenig Speicher dazu bei, die Größe der laufenden Prozesse zu verringern.

z*Faktor*
Multiplikator für Prioritätserhöhungen. Hiermit wird bestimmt, wieviel Gewicht auf den `Precedence`-Header gelegt wird. Der Default von **sendmail** ist 1800.

Z*Zahl* Die Priorität von Elementen in der Warteschlange um *Zahl* erhöhen, nachdem jeder Job verarbeitet wurde. **sendmail** verwendet einen Default von 90000.

Von sendmail verwendete Dateien

/usr/lib/sendmail
Das Binärprogramm von **sendmail**.

/usr/bin/newaliases
Ein Link auf */usr/lib/sendmail*; veranlaßt, daß die Aliasdatenbank neu aufgebaut wird.

/usr/bin/mailq
Ausgeben des Inhalts der Nachrichtenwarteschlange.

/etc/sendmail.cf
Konfigurationsdatei in Textform.

/etc/sendmail.hf
SMTP-Hilfedatei.

/usr/lib/sendmail.st
Statistikdatei. Muß nicht vorhanden sein.

/etc/aliases
Aliasdatei in Textform.

/etc/aliases.{pag,dir}
: Aliasdateien in **dbm**-Format.

/var/spool/mqueue
: Verzeichnis, in dem die Warteschlange und temporäre Dateien abgelegt werden.

*/var/spool/mqueue/qf**
: Kontrolldateien der Nachrichten.

*/var/spool/mqueue/df**
: Datendateien der Nachrichten.

/var/spool/mqueue/lf
: Lockdateien.

/var/spool/mqueue/tf
: Temporäre Kopien der *qf**-Dateien. Werden verwendet, wenn die Warteschlangendatei neu aufgebaut wird.

/var/spool/mqueue/nf
: Wird benutzt, wenn eine eindeutige ID gesucht wird.

/var/spool/mqueue/xf
: Protokoll der aktuellen Sitzung.

showmount [*Optionen*] [*Rechner*]

NFS/NIS-Befehl. Zeigt die Clients an, die Dateisysteme von *Rechner* (oder vom lokalen Rechner, falls kein Rechner angegeben ist) bei sich eingehängt haben.

Optionen

-a, --all
: Rechnername des Clients und Name des eingehängten Verzeichnisses anzeigen.

-d, --directories
: Nur eingehängte Verzeichnisse anzeigen, nicht die Clients.

-e, --exports
: Anzeigen der Verzeichnisse, die exportiert werden dürfen, und der Rechner, die diese importieren dürfen.

shutdown [*Optionen*] *Wann* [*Nachricht*]

Jede Programmabarbeitung beenden. *Wann* kann eine bestimmte Zeit (im Format *hh:mm*), eine Anzahl von Minuten, die gewartet wird (im Format +*m*), oder **now** sein. Eine *Nachricht* an alle Benutzer ruft diese auf, sich abzumelden. An alle Prozesse wird das Signal SIGTERM geschickt, damit sie sich ordnungsgemäß beenden können. Das eigentliche Herunterfahren wird durch */etc/init* ausgeführt. Dabei wird das System in den Runlevel 1 versetzt. Nur der Superuser kann den **shutdown**-Befehl ausführen. Nachrichten an alle Benutzer (Default oder angegeben) werden während der Wartezeit in regelmäßigen Abständen geschickt; je näher der Zeitpunkt des Herunterfahrens rückt, je öfter.

→

shutdown ←

Optionen

- -c Prozeß des Herunterfahrens (z.B. während der Wartezeit) abbrechen.
- -f Schnelles Neustarten, indem der übliche Aufruf von **fsck** übergangen wird.
- -h Anhalten des Systems, wenn das Herunterfahren abgeschlossen ist.
- -k Ausgeben der Warnung, aber das eigentliche Herunterfahren nicht durchführen.
- -n Herunterfahren oder Aufruf von **init** vornehmen.
- -r System nach erfolgtem Herunterfahren neu starten.
- -t *Sek* Mindestens *Sek* Sekunden zwischen dem Beenden der Prozesse und dem Wechsel des Runlevels warten.

slattach

slattach [*Optionen*] [*Terminal*]

TCP/IP-Befehl. Serielle Leitungen als Netzwerkschnittstellen konfigurieren, womit sie für die Verwendung als Point-to-Point-Verbindung vorbereitet sind. Nur der Superuser kann eine Netzwerkschnittstelle mit einer seriellen Leitung verbinden oder die Verbindung auflösen.

Optionen

- -c *Befehl*
  Befehl ausführen, wenn die Verbindung getrennt wird.
- -d Debugging-Modus.
- -e Programm sofort beenden, nachdem das Gerät initialisiert worden ist.
- -h Beenden, wenn die Verbindung getrennt wird.
- -l UUCP-ähnliche Lockdateien in */var/spool/uucp* anlegen.
- -L Dreidrahtverbindungen zulassen.
- -m Leitung nicht in den 8-Bit-Modus versetzen.
- -n Ähnlich wie **mesg n**.
- -p *Protokoll*
  Angeben des zu verwendenden Protokolls. *Protokoll* kann **slip**, **adaptive**, **ppp** oder **kiss** sein.
- -q Stiller Modus, keine Nachrichten ausgeben.
- -s *Geschwindigkeit*
  Leitungsgeschwindigkeit angeben.

swapdev

swapdev [*Gerät*] [*Image* [*Auslagerungsgerät* [*Position*]]]

Wenn keine Argumente angegeben werden, werden Benutzungsinformationen über das Auslagerungsgerät ausgegeben. Wenn lediglich die Lage des Kernel*images* angegeben wird, werden die dort vorgefundenen Informationen ausgegeben. Um diese Information zu verändern, muß ein neues Auslagerungsgerät angegeben werden. Außerdem kann eine *Position* im Kernel*image* angegeben werden, an der Informationen verändert werden sollen. **rdev -s** ist ein Synonym für **swapdev**.

swapdev

Option

-o *Position*

Wie *Position* als Argument.

swapon

swapon [*Optionen*] *Gerät*

Gerät (kann auch eine durch Leerzeichen getrennte Liste sein) als Auslagerungsgerät verwenden.

Optionen

-a
: In */etc/fstab* nach Geräten suchen, die mit **sw** markiert sind, und diese anstelle von *Gerät* verwenden.

-p *Priorität*
: Gibt die *Priorität* des Auslagerungsgerätes an. Auslagerungsbereiche mit höherer Priorität werden vor solchen mit niedrigerer Priorität verwendet.

swapoff

swapoff [*Option*] | *Gerät*

Gerät (kann auch eine durch Leerzeichen getrennte Liste sein) nicht mehr als Auslagerungsgerät verwenden.

Option

-a
: In */etc/fstab* nach Geräten suchen, die mit **sw** markiert sind, und diese anstelle von *Gerät* verwenden.

sync

sync

Schreibt die Buffer der Dateisysteme auf die Festplatte. **sync** führt den **sync()**-Systemaufruf aus. Wenn das System angehalten werden soll, muß **sync** aufgerufen werden, um die Integrität des Dateisystems zu gewährleisten. **shutdown** ruft **sync** automatisch auf, bevor das System heruntergefahren wird. **sync** kann einige Sekunden benötigen. Daher sollte für eine kurze Frist **sleep** aufgerufen werden, bevor **halt** oder **reboot** aufgerufen wird. Beachten Sie bitte, daß das System immer mit **shutdown** heruntergefahren oder neu gestartet werden sollte, da **shutdown** sich um das Aufrufen von **sync** und anderen Verwaltungsaufgaben kümmert.

SysAdmin Befehle

syslogd

syslogd

TCP/IP-Befehl. Protokolliert Systemmeldungen in einer Reihe von Dateien, die in der Konfigurationsdatei */etc/syslog.conf* angegeben werden. Jede Meldung besteht aus einer Zeile. Eine Meldung kann einen Prioritätscode enthalten, der durch eine Zahl in spitzen Klammern am Anfang der Zeile angegeben wird. Die Prioritäten werden in *<sys/syslog.h>* definiert. **syslogd** liest aus den Internet- Domänensockets, die in */etc/services* angegeben sind. Um **syslogd** zu beenden, schicken Sie ihm ein SIGTERM-Signal.

sysklogd

sysklogd

syslogd ist das Linux-Programm, das die Funktionalität von **syslogd** bereitstellt und sich genauso wie die BSD-Version von **syslogd** verhält. Der Unterschied sollte für den Benutzer nicht sichtbar sein. Allerdings ist **sysklogd** völlig anders programmiert und unterstützt auch eine etwas erweiterte Syntax. Der Dämon wird als **syslogd** aufgerufen. Siehe auch **klogd**.

Optionen

-d
: Debugging einschalten.

-f *Konfigurationsdatei*
: Andere Konfigurationsdatei angeben.

-h
: Nachrichten von entfernten Rechnern an weiterleitende Rechner weiterleiten.

-l *Rechnerliste*
: Rechner angeben, die nur mit ihrem Rechnernamen, aber nicht mit ihrem voll-qualifizierten Domänennamen aufgeführt werden. Wenn mehrere Rechner angegeben werden, müssen die Namen durch Doppelpunkte getrennt werden.

-m *Markierungsintervall*
: Abstand zwischen zwei Zeitmarkierungen auswählen.

-n
: Nicht automatisch in den Hintergrund gehen. Wird benötigt, wenn **syslogd** von **init** gestartet wird.

-p *Socket*
: Schickt das Protokoll an *Socket* statt an */dev/log*.

-r
: Nachrichten vom Netzwerk über ein Internet-Domänensocket mit dem **syslog**-Dienst empfangen.

-s *Domänenliste*
: In *Domänenliste* angegebene Domänennamen vor dem Protokollieren entfernen. Wenn mehrere Domänennamen angegeben werden, sollten sie durch einen Doppelpunkt getrennt werden.

systat

systat [*Optionen*] *Rechner*

Informationen über das Netzwerk oder den Systemzustand eines entfernten Rechners abfragen, indem die **netstat**-, **systat**- und **daytime**-Dienste des Rechners benutzt werden.

Optionen

-n, **--netstat**
: Nur den **netstat**-Dienst befragen.

-p *Port*, **--port** *Port*
: Bei der Anfrage zu verwendender Port.

-s, **--systat**
: Nur den **systat**-Dienst befragen.

-t, **--time**
: Nur den **daytime**-Dienst befragen.

talkd [*Option*]

TCP/IP-Befehl. Server für die Kommunikation zwischen entfernten Benutzern. **talkd** teilt einem Benutzer mit, daß jemand mit ihm eine Konversation beginnen möchte. Ein **talk**-Client beginnt ein Rendezvous durch Senden einer CTL_MSG-Nachricht des Typs LOOK_UP an den Server. Der Server durchsucht dann seine Einladungstabellen nach einer existierenden Gesprächseinladung für den Client. Wenn keine gefunden wird, schickt der Rufende eine ANNOUNCE-Nachricht, die dazu führt, daß der Server an alle Anmeldeterminals des Angerufenen eine Nachricht schickt und um eine Verbindung bittet. Wenn der Angerufene antwortet, antwortet der lokale Server wiederum mit der Rendezvous-Adresse, und es wird eine Stream-Verbindung aufgebaut, über welche die Konversation stattfindet.

Option

-d Debugging-Informationen via **syslogd** ausgeben.

tar [*Optionen*] [*Dateien*]

Kopieren von *Dateien* in ein oder aus einem Archiv. Wenn unter den *Dateien* Verzeichnisse sind, verarbeitet **tar** den gesamten Verzeichnisbaum darunter. Wenn kein Archiv mit der **-f**-Option angegeben wurde, wird die Standardein- bzw. -ausgabe verwendet. Der Bindestrich vor Optionen ist optional. Die einzige Ausnahme ist die Verwendung von langen Optionen wie **--modification-time**. In diesem Fall ist die genaue Syntax

tar --*lange-Option* -*Funktions-Optionen Dateien*

Beispiel:
```
tar --modification-time -xvf tararchiv.tar
```

Funktionsoptionen

Sie müssen genau eine dieser Optionen angeben, die vor allen anderen Optionen stehen muß:

-c, --**create**
: Neues Archiv anlegen.

-d, --**compare**
: Vergleichen der im Archiv abgelegten Dateien mit *Dateien*. Alle Unterschiede (z.B. fehlende Dateien, unterschiedliche Größen, unterschiedliche Dateiattribute wie Zugriffsrechte und Änderungszeiten) werden ausgegeben.

-r, --**append**
: *Dateien* an das Ende eines bereits existierenden Archives anhängen.

-t, --**list**
: Ausgeben der Namen von *Dateien*, wenn sie im Archiv abgespeichert sind; wenn *Dateien* nicht angegeben wird, wird der gesamte Inhalt des Archivs ausgegeben.

→

-u, --update

Dateien hinzufügen, wenn sie sich noch nicht im Archiv befinden oder sich geändert haben.

-x, --extract, --get

Dateien aus dem Archiv extrahieren (wenn *Dateien* nicht angegeben wird, alle Dateien extrahieren).

-A, --catenate, --concatenate

Ein zweites Archiv an das Ende des ersten anhängen.

Optionen

n Gerät Nummer *n* auswählen, wobei *n* zwischen 0 und 9999 liegen muß. Der Default steht in */etc/default/tar*.

[*Laufwerk*][*Dichte*]

(Band-)*Laufwerk* (0–7) und (Schreib-)*Dichte* (**l**, **m** oder **h** für niedrige, mittlere oder hohe Schreibdichte) angeben.

--atime-preserve

Ursprüngliche Zugriffszeit bei den extrahierten Dateien erhalten.

-b, --block-size=*n*

Blockgröße auf $n \times 512$ Bytes setzen.

--checkpoint

Angetroffene Verzeichnisnamen ausgeben.

--exclude=*Datei*

Datei aus jeder Dateiliste entfernen.

-f *Arch*, **--file=***Dateiname*

Archiv *Arch* verwenden. *Dateiname* kann auch die Form *Rechnername:Dateiname* haben.

--force-local

Dateinamen der Form *Rechnername:Dateiname* als lokale Dateien interpretieren.

-g, --listed-incremental

Inkrementelle Sicherung des neuen Typs anlegen.

-h, --dereference

Symbolische Links dereferenzieren.

-i, --ignore-zeros

Blöcke der Größe Null (d.h. *EOF*s) ignorieren.

--ignore-failed-read

Unlesbare Dateien bei der Archivierung ignorieren. Das Defaultverhalten ist ein Programmabbruch.

-k, --keep-old-files

Wenn Dateien extrahiert werden, Dateien mit ähnlichen Namen nicht überschreiben. Statt dessen wird eine Fehlermeldung ausgegeben.

-l, --one-file-system

Keine Dateien aus anderen Dateisystemen archivieren.

-m, --modification-time

Modifizierungszeiten nicht restaurieren; sie werden auf den Zeitpunkt der Extraktion gesetzt.

--null Dateinamen, die via **-T** gelesen werden, dürfen mit Null abgeschlossen werden. Überschreibt **-C**.

--old, --portability, --preserve
Äquivalent zu **-p** und **-s**.

-p, --same-permissions, --preserve-permissions
Besitzer- und Gruppen-ID bleiben so wie im Original.

--remove-files
Originaldateien nach Aufnahme im Archiv löschen.

--rsh-command=*Befehl*
Nicht mit **rsh** zu einem entfernten Rechner verbinden, sondern statt dessen *Befehl* verwenden.

-s, --same-order, --preserve-order
Beim Extrahieren Dateinamen nach der Reihenfolge im Archiv sortieren.

--totals
Bytesummen ausgeben.

--use-compress-program=*Programm*
Archivierte Dateien mit *Programm* komprimieren und extrahierte Dateien mit *Programm* dekomprimieren.

-v, --verbose
Verbose-Modus. Dateinamen ausgeben, die einem Archiv hinzugefügt oder aus einem Archiv extrahiert werden.

-w, --interactive
Jede Aktion vom Benutzer mit **y** bestätigen lassen.

-z, --gzip, --ungzip
Dateien vor der Archivierung mit **gzip** komprimieren und vor der Extraktion mit **gunzip** dekomprimieren.

-C, --directory=*Verzeichnis*
Vor der Befehlsausführung nach *Verzeichnis* wechseln.

-F, --info-script, --new-volume-script=*Skript*
Setzt **-M** voraus (mehrere Archivdateien). Am Ende jeder Archivdatei wird *Skript* ausgeführt.

-G, --incremental
Inkrementelle Sicherungen des alten Typs anlegen.

-K *Datei*, **--starting-file** *Datei*
tar-Operation mit *Datei* im Archiv beginnen.

-L, --tape-length=*Länge*
Höchstens *Länge* × 1024 Bytes auf jedes Band schreiben.

-M, --multi-volume
Multi-Volume-Archiv auslesen (oder mit **-c** anlegen).

-N *Datum*, **--after-date** *Datum*
Dateien ignorieren, die älter als *Datum* sind.

-O, --to-stdout
Extrahierte Dateien auf die Standardausgabe ausgeben.

-P, --absolute-paths
Keine Schrägstriche am Anfang von Dateinamen entfernen.

→

tar
←

-R, --**record-number**
Datensatznummer des Archivs anzeigen.

-S, --**sparse**
Kurze Dateien auf eine besondere, effizientere Weise behandeln.

-T *Datei*, --**files-from** *Datei*
Namen der zu archivierenden oder zu extrahierenden Dateien *Datei* entnehmen.

-V *Name*, --**label**=*Name*
Dieses Volume mit *Name* benennen.

-W, --**verify**
Archiv nach dem Anlegen auf Integrität überprüfen.

-X *Datei*, --**exclude** *Datei*
Die in *Datei* enthaltenen Dateien nicht archivieren oder extrahieren.

-Z, --**compress**, --**uncompress**
Dateien mit **compress** vor der Archivierung komprimieren und mit **uncompress** vor der Extraktion dekomprimieren.

Beispiele

Ein Archiv mit den Verzeichnisbäumen */bin* und */usr/bin* anlegen (**c**), den Fortschritt anzeigen (**v**) und das Archiv auf dem Bandlaufwerk */dev/rmt0* ablegen.

```
tar cvf /dev/rmt0 /bin /usr/bin
```

Den Inhalt des Bandes in einem Format ähnlich wie**ls -l** anzeigen:

```
tar tvf /dev/rmt0
```

Das */bin*-Verzeichnis extrahieren:

```
tar xvf /dev/rmt0 /bin
```

Ein Archiv des aktuellen Verzeichnisses anlegen und in der Datei *backup.tar* speichern:

```
tar cvf - `find . -print` > backup.tar
```

Der einzelne Bindestrich teilt **tar** mit, daß das Archiv auf die Standardausgabe geschrieben werden soll. Diese wird dann in die Datei *backup.tar* umgeleitet.

tcpd

tcpd

TCP/IP-Befehl. Ein Dämon, der eingehende TCP/IP-Anfragen (wie z.B. von **telnet**, **ftp**, **finger**, **exec** oder **rlogin**) überwacht. Sämtliche Dienste können überprüft und protokolliert werden, bevor die Anfragen an den entsprechenden Dämon weitergereicht werden.

tcpdchk [*Optionen*]

tcpdchk

TCP/IP-Befehl. Liest die Konfigurationsdateien des TCP-Wrappers ein (*etc/hosts.allow* und */etc/hosts.deny*), gibt eine Liste der möglichen Probleme aus und macht Vorschläge, wie diese zu beheben sind.

Optionen

-a
: Eine Regelliste mit ausgeben. Es ist kein ALLOW-Schlüsselwort erforderlich, bevor von anderen Netzwerken auf den lokalen Rechner zugegriffen werden darf.

-d
: */etc/hosts.allow* und */etc/hosts.deny* einlesen.

-i *Konfigurationsdatei*
: Gibt den Ort der Dateien *inetd.conf* und *tlid.conf* an. Diese Dateien werden von **tcpdchk** bei der Überprüfung der TCP-Wrapperdateien verwendet.

-v
: Verbose-Modus.

tcpdmatch [*Optionen*] *Dämon Client*

tcpdmatch

TCP/IP-Befehl. Sagt die Antwort des TCP-Wrappers auf eine bestimmte Anfrage voraus. Sie müssen angeben, an welchen *Dämon* sich die Anfrage richtet und von welchem *Client* sie ausgeht (als Syntax kann *Benutzer@Client* oder ein Jokerzeichen verwendet werden). Die Aktionen des Wrappers werden aus */etc/hosts.allow* und */etc/hosts.deny* bestimmt.

Optionen

-d
: */etc/hosts.allow* und */etc/hosts.deny* einlesen.

-i *Konfigurationsdatei*
: Gibt den Ort der Dateien *inetd.conf* und *tlid.conf* an. Diese Dateien werden von **tcpmatch** bei der Überprüfung der TCP-Wrapperdateien verwendet.

telinit [*Option*] [*Runlevel*]

telinit

Teilt **init** mit, daß der Runlevel des Systems gewechselt werden soll. **telinit** ist tatsächlich nur ein Link auf den **init**-Befehl.

Option

-t *Sekunden*
: Schickt das SIGKILL-Signal *Sekunden* Sekunden nach SIGTERM. Der Default ist 20.

Runlevel

Die Default-Runlevels unterscheiden sich von Distribution zu Distribution, aber die folgenden sind standardisiert:

0
: System anhalten.

1, s, S
: Single-User-Modus (Wartungsmodus).

→

SysAdmin Befehle

telinit ←

6	System neu starten.
a, b, c	Nur die Einträge in */etc/inittab* verarbeiten, die mit den Runlevels **a**, **b** oder **c** markiert sind.
q, Q	*/etc/inittab* neu einlesen.

Schauen Sie in der Datei */etc/inittab* nach, um die auf Ihrem System gültigen Runlevels zu erfahren.

---

telnetd

telnetd [*Optionen*]

TCP/IP-Befehl. TELNET Protocol Server. **telnetd** wird von **inetd** bei Anfragen am TELNET-Port (defaultmäßig 23) aufgerufen. **telnetd** legt für den Client ein Pseudoterminal an und startet damit einen **login**-Prozeß, der mit **stdin**, **stdout** und **stderr** am Pseudoterminal verbunden ist. **telnetd** manipuliert die Master-Seite des Pseudoterminals, indem das TELNET-Protokoll implementiert wird und Zeichen zwischen dem entfernten Client und dem **login**-Prozeß hin und her geschickt werden.

Optionen

-debug [*Port*]
: Startet **telnetd** manuell anstatt durch **inetd**. *Port* kann als alternativer TCP-Port angegeben werden, auf dem **telnetd** arbeiten soll.

-D *Modifikator(en)*
: Debugging-Modus. **telnet** gibt damit Debugging-Informationen über die Verbindung aus. Damit kann der Benutzer sehen, was **telnet** tut. Eine Reihe von Modifikatoren steht für den Debugging-Modus zur Verfügung:

exercise	Noch nicht implementiert.
netdata	Von **telnetd** empfangenen Datenstrom anzeigen.
options	Informationen über das Aushandeln der TELNET-Optionen anzeigen.
ptydata	Daten anzeigen, die auf das Pseudoterminal geschrieben werden.
report	Wie **options**, außerdem Ausgeben von Informationen, welche Verarbeitung gerade stattfindet.

---

tftpd

tftpd [*Heimatverzeichnis*]

TCP/IP-Befehl. Trivial File Transfer Protocol Server. **tftpd** wird normalerweise von **inetd** gestartet und arbeitet dann auf dem Port, der in der **tftp**-Beschreibung der Datei */etc/inetd.conf* angegeben ist. Defaultmäßig ist der Eintrag für **tftp** in der Datei */etc/inetd.conf* auskommentiert; um **tftp** verwenden zu können, muß das Kommentarzeichen entfernt werden. Bevor auf eine Anfrage geantwortet wird, versucht der Server, sein aktuelles Verzeichnis in *Heimatverzeichnis* abzuändern. Wenn dieses nicht angegeben ist, wird *tftpboot* angenommen.

traceroute [*Optionen*] *Rechner* [*Paketgröße*]

traceroute

TCP/IP-Befehl. Verfolgt die Route, die ein Paket nimmt, um einen bestimmten Rechner zu erreichen. Dies geschieht, indem **traceroute** UDP-Pakete mit einer kleinen Restlebenszeit (TTL) losschickt und auf ICMP „time exceeded"-Nachrichten vom Gateway wartet. *Rechner* ist der Name oder die IP-Nummer des gesuchten Rechners. *Paketgröße* ist die Paketgröße des Test-Datagramms in Bytes. Der Default ist 38 Bytes.

Optionen

-d
: Debuggen auf Socket-Ebene einschalten.

-g *Adr*
: Neben den TTL-Tests auch noch die IP-LSRR (Loose Source Record Route-) Option einschalten. Damit kann festgestellt werden, wie jemand an der IP-Adresse *Adr* ein bestimmtes Ziel erreichen kann.

-l
: Restlebenszeit für jedes empfangene Paket angeben.

-m *Max-TTL*
: Maximale Restlebenszeit, die in ausgehenden Testpaketen verwendet wird. Der Default ist 30 Hops.

-n
: Hop-Adressen numerisch anstatt symbolisch ausgeben.

-p *Port*
: UDP-Portnummer für Testpakete auf *Port* setzen. Der Default ist 33434.

-q *n*
: Einstellen der Anzahl der Testpakete für jeden Restlebenszeitwert auf *n*. Der Default ist 3.

-r
: Normale Routing-Tabellen umgehen und die Pakete direkt an einen Rechner eines angeschlossenen Netzwerks senden.

-s *Quell-Adr*
: *Quell-Adr* als Quelladresse in ausgehenden IP-Paketen eintragen.

-t *Tos*
: Einstellen der Dienstart (*type-of-service*) in ausgehenden Testpaketen auf *Tos*. Der Default ist 0. *Tos* muß eine dezimale Ganzzahl zwischen 0 und 255 sein.

-v
: Verbose-Modus. Alle ICMP-Pakete (außer TIME_EXCEEDED und PORT_UNREACHABLE) werden angezeigt.

-w *Wartezeit*
: Wartezeit für eine Antwort auf ein ausgehendes Testpaket auf *Wartezeit* Sekunden setzen. Der Default ist 3 Sekunden.

SysAdmin Befehle

tune2fs [*Optionen*] *Gerät*

tune2fs

Feineinstellung der Parameter eines Linux-Second-Extended-Dateisystems durch Anpassen diverser Parameter. Sie müssen das Gerät angeben, auf dem das Dateisystem liegt. Das Dateisystem darf nicht schreibbar eingehängt sein, wenn Parameter verändert werden sollen.

→

tune2fs
←

Optionen

-c *Einhängezähler*
: Angeben der maximalen Anzahl an Einhängevorgängen (mount counts) zwischen zwei Überprüfungen des Dateisystems.

-e *Verhalten*
: Angeben, wie sich der Kernel bei Fehlern im Dateisystem verhalten soll. *Verhalten* muß eines der folgenden Schlüsselwörter sein:

continue Normal weitermachen.

remount-ro Das fehlerhafte Dateisystem in nicht-schreibbarem Modus erneut einhängen.

panic Erzeugen einer Kernel Panic.

-g *Gruppe*
: Mitglieder von *Gruppe* (kann numerisch oder als Name angegeben werden) dürfen reservierte Blöcke verwenden.

-i *Intervall* [d|w|m]
: Angeben des maximalen Intervalls zwischen zwei Überprüfungen des Dateisystems. Verwendbare Einheiten sind **d** (Tage), **w** (Wochen) und **m** (Monate). Wenn *Intervall* 0 ist, ist das Überprüfen dieses Dateisystems überhaupt nicht zeitabhängig.

-l Eine Liste mit dem Inhalt des Superblocks ausgeben.

-m *Prozentsatz*
: Anteil der Blöcke, die für den Superuser reserviert sind, angeben.

-r *Anzahl*
: Anzahl der Blöcke angeben, die für den Superuser reserviert werden.

-u *Benutzer*
: *Benutzer* (kann numerisch oder als Name angegeben werden) darf reservierte Blöcke belegen.

---

tunelp

tunelp *Gerät* [*Optionen*]

Einstellen der Geräteparameter eines Druckers. Ohne Optionen werden Informationen über das Gerät ausgegeben.

Optionen

-a [on|off]
: Gibt an, ob das Programm abgebrochen werden soll, wenn ein Druckerfehler auftritt. Defaultmäßig wird nicht abgebrochen.

-c *n* **n**-mal erneut versuchen, das Gerät anzusprechen, wenn es ein Zeichen nicht annimmt. (Der Default ist 250.) Wenn *n* erreicht ist, wird pausiert, bevor ein neuer Versuch unternommen wird.

-i *IRQ* *IRQ* für den angegebenen Parallelport verwenden. -t und -c ignorieren. Wenn *IRQ* 0 ist, den nicht durch Interrupts getriebenen Zustand (Polling) wiederherstellen.

-o [on|off]
: Angeben, ob abgebrochen werden soll, wenn ein Gerät nicht erreichbar oder ohne Papier ist.

-q [on|off]
: Schaltet das Ausgeben der aktuellen IRQ-Einstellungen ein oder aus.

-r Port zurücksetzen.

-s Aktuellen Status des Druckers ausgeben.

-t *Zeit* Gibt eine Verzögerungszeit in Jiffies an, die gewartet werden soll, bevor ein zurückgewiesenes Zeichen noch einmal gesendet wird. Ein Jiffy ist definiert entweder als ein Tick der Systemuhr oder ein Phasendurchlauf des Wechselstroms; es sollte etwa eine Hundertstelsekunde sein.

-w *Zeit*
: Gibt die Verzögerung in Jiffies an, bevor ein Strobe-Signal erneut gesendet wird.

-C [on|off]
: Extrem vorsichtiges Überprüfen des Druckers auf Fehler ein- oder ausschalten.

tunelp

umount [*Optionen*] [*Gerätedatei/Verzeichnis*]

umount

Aushängen eines Dateisystems. **umount** teilt dem System mit, daß ein entfernbares Dateisystem, das vorher über die *Gerätedatei* eingehängt war, ausgehängt werden soll. Statt der *Gerätedatei* kann auch das *Verzeichnis* angegeben werden. Alle noch ausstehenden E/A-Operationen werden abgeschlossen und das Dateisystem als sauber markiert.

Optionen

-a Alle Dateisysteme aushängen, die in */etc/mtab* aufgeführt sind.

-n Aushängen, aber */etc/mtab* nicht aktualisieren.

-t *Typ* Nur Dateisysteme des Typs *Typ* aushängen.

update [idletasks]

update

Alle ausstehenden Ausgaben auf den Monitor ausgeben, warten, daß der Server die Ausgaben verarbeitet, alle ausstehenden Ereignisse verarbeiten, dann noch einmal alle ausstehenden Ausgaben auf den Monitor ausgeben. **update** wiederholt diesen Vorgang, bis keine Ereignisse oder Idle-Handler mehr ausstehen.

Wenn das Schlüsselwort **idletasks** angegeben wird, werden keine neuen Ereignisse oder Fehler mehr verarbeitet; lediglich die Idle-Handler werden noch ausgeführt.

vidmode

vidmode [*Option*] *Image* [*Modus* [*Position*]]

Einstellen des Videomodus für den Kernel *Image*. Wenn keine Argumente angegeben wurden, wird der aktuelle Modus ausgegeben. *Modus* ist ein ein Byte großer Wert, der an Position 506 im Kernelimage liegt. Der Modus kann verändert werden, indem das zu verändernde Kernel*image*, der neue *Modus* sowie die Position im Kernelimage angegeben werden. Die Position kann auch weggelassen werden, der Default ist 506. **rdev -v** ist ein Synonym für **vidmode**. Wenn **LILO** verwendet wird, wird **vidmode** nicht gebraucht. In diesem Fall kann der Videomodus auch von der **LILO**-Eingabeaufforderung während des Systemstarts angegeben werden.

Modi

- -3 Benutzer fragen
- -2 Erweitertes VGA
- -1 Normales VGA
- 0 Als ob **0** am Eingabeprompt eingegeben worden wäre
- 1 Als ob **1** am Eingabeprompt eingegeben worden wäre
- 2 Als ob **2** am Eingabeprompt eingegeben worden wäre
- 3 Als ob **3** am Eingabeprompt eingegeben worden wäre
- n Als ob **n** am Eingabeprompt eingegeben worden wäre

Option

-o *Position*
Wie *Position* als Argument

w

w [*Optionen*] [*Benutzer*]

Ausgeben von Zusammenfassungen der Systemnutzung, der aktuell angemeldeten Benutzer und dessen, was diese gerade tun. **w** ist im Prinzip nur eine Kombination von **uptime**, **who** und **ps -a**. Durch Angabe von *Benutzer* kann die Ausgabe auf einen Benutzer eingeschränkt werden.

Optionen

- -h Überschriften und **uptime**-Informationen nicht ausgeben.
- -i Nach der Zeit, in der kein Befehl eingegeben wurde, sortieren.
- -n IP-Adressen nicht in Namen umsetzen.

Dateien

/var/run/utmp
Liste der aktuell angemeldeten Benutzer.

wall

wall [*Datei*]

An alle Benutzer schreiben. **wall** liest eine Mitteilung von der Standardeingabe bis zum Dateiendekennzeichen. Danach wird diese Mitteilung an alle gerade angemeldeten Benutzer geschickt, wobei `Broadcast`

`Message from...` vorangestellt wird. Wenn *Datei* angegeben wird, wird die Mitteilung hieraus statt von der Standardeingabe entnommen. | wall

ypbind [*Optionen*] ypbind

NFS/NIS-Befehl. NIS-Bindeprozeß. **ypbind** ist ein Dämon, der in der Regel während des Hochfahrens des Systems aus der Datei */etc/nfs* gestartet wird. Seine Aufgabe besteht darin, Informationen zu speichern, mit denen Client-Prozesse auf einem Rechner mit einem **ypserv**-Prozeß kommunizieren können. Die von **ypbind** gespeicherten Informationen werden *Bindung* genannt — die Verbindung eines Domänennamens mit der Internet-Adresse des NIS-Servers und dem Port auf diesem Rechner, an dem der **ypserv**-Prozeß auf Anfragen wartet. Diese Information wird in der Datei */var/yp/bindings/Domänename.Version* gespeichert.

-ypset Kann verwendet werden, um eine Bindung zu ändern. Diese Option ist sehr gefährlich und sollte nur verwendet werden, um Fehler im Netzwerk von einem entfernten Rechner aus zu suchen.

-ypsetme
: **ypset**-Anfragen dürfen nur von diesem Rechner aus gesendet werden. Diese Sicherheit basiert auf dem Überprüfen der IP-Adresse. Diese kann aber in Netzwerken kompromittiert werden, in denen unzuverlässige oder böswillige Benutzer Pakete einschleusen. Diese Option sollte daher nicht verwendet werden.

ypcat [*Optionen*] *Tabellenname* ypcat

NFS/NIS-Befehl. Ausgeben der Werte in einer NIS-Datenbank, die durch *Tabellenname* angegeben wird. *Tabellenname* kann entweder ein Tabellenname oder ein Tabellenkurzname sein.

Optionen

-d *Domäne*
: Eine andere Domäne als die Defaultdomäne angeben.

-k Schlüssel in der Tabelle ausgeben, bei denen der Wert Null oder der Schlüssel kein Bestandteil des Wertes ist.

-t *Tabellenname* nicht in einen Tabellennamen übersetzen.

-x Ausgeben der Kurznamentabelle, welche die bekannten Kurznamen und zugehörigen Tabellennamen ausgibt. Erfordert kein *Tabellenname*-Argument.

ypinit [*Optionen*] ypinit

NFS/NIS-Befehl. Aufbauen und Installieren einer NIS-Datenbank auf einem NIS-Server. **ypinit** kann verwendet werden, um einen Master- oder einen Slave-Server oder einen Slave-Copier zu installieren. Nur der Superuser kann **ypinit** ausführen.

→

SysAdmin Befehle

ypinit ←

Optionen

-c *Mastername*
: Installieren einer Slave-Copier-Datenbank. *Mastername* sollte der Rechnername eines NIS-Servers sein: Entweder der Master-Server für alle Tabellen oder ein Server, auf dem die Datenbank aktuell und stabil ist.

-m
: Gibt an, daß der lokale Rechner der NIS-Server sein soll.

-s *Mastername*
: Installieren einer Slave-Server-Datenbank. *Mastername* sollte der Rechnername eines NIS-Servers sein: Entweder der Master-Server für alle Tabellen oder ein Server, auf dem die Datenbank aktuell und stabil ist.

---

ypmatch

ypmatch [*Optionen*] *Schlüssel*... *Tabellenname*

NFS/NIS-Befehl. Ausgeben eines oder mehrerer *Schlüssel* einer NIS-Tabelle *Tabellenname*. Der Tabellenname kann auch ein Tabellenkurzname sein.

Optionen

-d *Domäne*
: *Domäne* anstelle der Defaultdomäne verwenden.

-k
: Vor jedem Wert eines Schlüssels den Schlüssel selbst und einen Doppelpunkt ausgeben.

-t
: Tabellenkurznamen nicht in Tabellennamen übersetzen.

-x
: Ausgeben der Kurznamentabelle, welche die bekannten Kurznamen und zugehörigen Tabellennamen ausgibt. Erfordert kein *Tabellenname*-Argument.

---

yppasswd

yppasswd [*Name*]

NFS/NIS-Befehl. Anmeldepaßwort im Network Information Service (NIS) ändern. **yppasswd** erfragt das alte Paßwort sowie ein neues. Das neue Paßwort muß mindestens vier Zeichen lang sein, wenn eine Mischung aus Klein- und Großbuchstaben verwendet wird, oder sechs Zeichen, wenn nur Klein- oder nur Großbuchstaben verwendet werden.

---

yppasswdd

rpc.yppasswdd [*Option*]

NFS/NIS-Befehl. Server zur Bearbeitung der NIS-Paßwortdatei. **yppasswdd** bearbeitet die von **yppasswd** gestellten Anfragen, ein Paßwort zu ändern. Das Paßwort wird dabei nur geändert, wenn das von **yppasswd** geschickte alte Paßwort dem Paßwort des Eintrags entspricht und die Benutzer- und Gruppen-ID identisch mit denen in der Datei */etc/passwd* des Servers sind. **yppasswdd** aktualisiert dann die Datei */etc/passwd* und die Paßworttabellen des lokalen Servers.

Option

-s
: Unterstützung von Shadow-Paßwörtern.

yppoll [*Optionen*] *Tabellenname*

yppoll

NFS/NIS-Befehl. Feststellen der Version einer NIS-Tabelle auf einem NIS-Server. **yppoll** befragt den **ypserv**-Prozeß nach einer laufenden Nummer und dem Rechnernamen des Master-NIS-Servers für die angegebene Tabelle.

Optionen

-h *Rechner*
: **ypserv**-Prozeß auf dem angegebenen *Rechner* nach den Tabellenparametern fragen. Wenn der *Rechner* nicht angegeben ist, wird der NIS-Server des lokalen Rechners (der mit **ypwhich** abgefragt werden kann) verwendet.

-d *Domäne*
: *Domäne* anstelle der Defaultdomäne verwenden.

yppush [*Optionen*] *Tabellennamen*

yppush

NFS/NIS-Befehl. Weiterverteilen der geänderten NIS-Tabellen erzwingen. **yppush** kopiert eine neue Version der angegebenen NIS-Tabellen vom Master-NIS-Server auf die Slave-NIS-Server. Dazu legt es zunächst eine Liste von NIS-Serverrechnern an, indem die Tabelle **ypservers** mit der Option -d und einer *Domäne* eingelesen wird. Die Schlüssel dieser Tabelle sind die ASCII-Namen der Rechner, auf denen die NIS-Server laufen. An den NIS-Server jedes Rechners wird eine „transfer map"-Anfrage geschickt, die unter anderem alle notwendigen Informationen enthält, damit das Transferprogramm **yppush** zurückrufen kann. Wenn die Übertragung erfolgreich abgeschlossen worden ist und das Transferprogramm an **yppush** eine Statusmeldung geschickt hat, werden die Ergebnisse auf die Standardausgabe ausgegeben. **yppush** wird normalerweise von */var/yp/Makefile* aufgerufen.

Optionen

-d *Domäne*
: Angeben einer Domäne.

-v
: Verbose-Modus — eine Nachricht ausgeben, wenn ein Server angesprochen worden ist oder geantwortet hat.

SysAdmin Befehle

ypserv [*Optionen*]

ypserv

NFS/NIS-Befehl. NIS-Serverprozeß. **ypserv** ist ein Dämon, der normalerweise beim Hochfahren des Systems gestartet wird. Er läuft nur auf NIS-Serverrechnern mit vollständigen NIS-Datenbanken. Seine Hauptfunktion besteht darin, Informationen in den lokalen NIS-Datenbanken nachzuschlagen. Die Arbeitsweise von **ypserv** ist für den Implementierer durch die NIS-Protokollspezifikation und für den Programmierer durch die Headerdatei *<rpcvc/yp_prot.h>* festgelegt. Die Kommunikation mit **ypserv** wird über RPC-Aufrufe durchgeführt.

→

ypserv ←

Optionen

-d Weitere Informationen über Rechner von DNS beziehen.

-localonly
: **ypserv** antwortet nicht auf Anfragen von außen.

Dateien und Verzeichnisse

/var/yp/[Domänenname]/
: Verzeichnis mit den NIS-Datenbanken.

/var/yp/Makefile
: Makefile zum Erzeugen von NIS-Datenbanken.

ypset

ypset [*Optionen*] *Server*

NFS/NIS-Befehl. Weist **ypbind** auf einen bestimmten Server hin. **ypset** teilt **ypbind** mit, daß die NIS-Dienste für die angegebene Domäne von *Server* bezogen werden sollen. *Server* ist ein NIS-Server, mit dem **ypbind** sich verbinden soll, und kann als Name oder als IP-Adresse ausgegeben werden.

Optionen

-d *Domäne*
: *Domäne* anstelle der Defaultdomäne verwenden.

-h *Rechner*
: Einstellen der Bindung von **ypbind** auf *Rechner* anstatt auf den lokalen Rechner. *Rechner* kann als Name oder als IP-Adresse angegeben werden.

ypwhich

ypwhich [*Optionen*] [*Rechner*]

NFS/NIS-Befehl. Gibt den Rechnernamen des NIS-Servers oder Tabellenmasters zurück. Ohne Argumente gibt **ypwhich** den NIS-Server für den lokalen Rechner aus. Wenn *Rechner* angegeben wird, wird dieser Rechner befragt, welchen NIS-Server er verwendet.

Optionen

-d *Domäne*
: *Domäne* anstelle der Defaultdomäne verwenden.

-m *Tabelle*
: Master-NIS-Server für eine Tabelle herausfinden. In diesem Fall kann kein *Rechner* angegeben werden. *Tabelle* kann ein Tabellenname oder ein Tabellenkurzname sein.

-t *Tabellenname*
: Auflösung der Tabellenkurznamen verhindern.

-x Tabelle der Tabellenkurznamen ausgeben. Keine anderen Optionen möglich.

ypxfr [*Optionen*] *Tabellenname*

ypxfr

NFS/NIS-Befehl. Transferieren einer NIS-Tabelle in der Defaultdomäne des lokalen Rechners auf den lokalen Rechner selbst. Dazu werden normale NIS-Dienste verwendet. **ypxfr** legt eine temporäre Tabelle in */etc/yp/Domäne* an (wobei *Domäne* die Defaultdomäne des lokalen Rechners ist), füllt sie mit den Tabelleneinträgen, holt die Tabellenparameter und lädt sie. Wenn **ypxfr** interaktiv gestartet wird, schreibt es seine Ausgabe auf das Terminal. Wenn es dagegen ohne kontrollierendes Terminal gestartet wird und die Logdatei */usr/admin/nislog* existiert, dann werden alle Ausgaben an diese Datei angehängt.

Optionen

-b
: Resolver-Schalter in der Tabelle während des Transfers erhalten.

-C *TID Prog IP-Adr Port*
: Nur zur Verwendung durch **ypserv**. Wenn **ypserv** **ypxfr** aufruft, teilt es mit, daß **ypxfr** einen **yppush**-Prozeß zurückrufen soll, der auf dem Rechner mit der IP-Adresse *IP-Adr* läuft, als Programm mit der Nummer *Prog* registriert ist, auf *Port* lauscht und auf eine Antwort auf die Transaktion Nummer *TID* wartet.

-c
: Keine Anfrage „Aktuelle Tabelle löschen“ an den lokalen **ypserv**-Prozeß schicken.

-d *Domäne*
: *Domäne* anstelle der Defaultdomäne verwenden.

-f
: Auch dann Tabellen transferieren, wenn die Version auf dem Master älter ist als die lokale Version.

-h *Rechner*
: Tabelle von *Rechner* holen, unabhängig davon, welchen Rechner die Tabelle für ihren Master hält. Wenn der Rechner nicht angegeben wird, fragt **ypxfr** den NIS-Dienst nach dem Namen des Masters und versucht, die Tabelle von dort zu bekommen. *Rechner* kann als Name oder als IP-Adresse angegeben werden.

-S
: Nur NIS-Server verwenden, die als **root** laufen und einen reservierten Port benutzen.

-s *Domäne*
: Angeben einer Quell*domäne*, aus der eine Tabelle übertragen wird, die in allen Domänen gleich sein soll (z.B. die *services.byname*-Tabelle).

zdump [*Optionen*] [*Zonen*]

zdump

Eine Liste aller bekannten Zeitzonen ausgeben. Wenn ein Argument angegeben wird, werden nur die darin enthaltenen Zonen ausgegeben. Bei jeder Zone wird die aktuelle Zeit mit angegeben.

→

zdump ←

Optionen

-c *Jahr*
: Ein Jahr angeben, das abgeschnitten werden soll, um ausführliche Ausgaben etwas zu verkürzen. Nur sinnvoll mit **-v**.

-v
: Verbose-Modus. Zusätzliche Informationen über jede Zone mit ausgeben.

zic

zic [*Optionen*] [*Dateien*]

Aus den angegebenen Dateien Informationsdateien zur Zeitkonvertierung erzeugen. Wenn - als Datei angegeben wird, werden die Informationen von der Standardeingabe gelesen.

Optionen

-d *Verzeichnis*
: Die neu angelegten Dateien im *Verzeichnis* ablegen. Der Default ist */usr/local/etc/zoneinfo.*

-l *Zeitzone*
: Angeben, welche *Zeitzone* für die lokale Zeit verwendet werden soll. **zic** erzeugt einen Link zwischen der Zoneninformation für *Zeitzone* und der Zone **localtime**.

-p *Zeitzone*
: Einstellen der Default-Regeln zur Behandlung von Umgebungsvariablen im POSIX-Format auf den durch *Zeitzone* angegebenen Zonennamen.

-s
: Zeitwerte nur dann abspeichern, wenn sie vorzeichenbehaftet und vorzeichenlos identisch sind.

-v
: Verbose-Modus. Zusätzliche Überprüfungen auf Fehler durchführen und Warnungen ausgeben.

-y *Befehl*
: Jahreszahlen mit *Befehl* überprüfen. Der Default ist **yearistype**.

-L *Datei*
: Informationen über Schaltsekunden aus *Datei* entnehmen.

Die Quelldateien für **zic** sollten als Folge von Regelzeilen, Zonenzeilen und Linkzeilen formatiert sein. Auf der Kommandozeile kann zusätzlich eine Datei angegeben werden, die Regeln für Schaltsekunden enthält. Regelzeilen beschreiben, wie Zeiten berechnet werden. Sie geben Zeitveränderungen, Sommerzeit, Kriegszeit und andere Veränderungen an, die eine bestimmte Zeitzone betreffen können. Zonenzeilen geben an, welche Regeln auf welche Zeitzonen zutreffen. Linkzeilen verbinden ähnliche Zonen miteinander. Schaltzeilen geben die genaue Zeit an, zu der Sekunden hinzugezählt bzw. von der Sekunden abgezogen werden sollen. Jede dieser Zeilen besteht aus einzelnen Feldern. Die Felder werden voneinander durch beliebig viele Leerzeichen getrennt. Kommentarzeilen beginnen mit einem #. Folgende Felder sind in allen Zeilen verwendbar:

Felder in Regelzeilen

`Rule` *`NAME VON BIS TYP IM AM UM SOMMER BUCHSTABEN`*

NAME
Name dieser Regel

VON
Erstes Jahr, in dem diese Regel gültig ist. Es wird der Gregorianische Kalender vorausgesetzt. Anstelle eines konkreten Jahres können auch **minimum** bzw. **maximum** für das kleinste bzw. größte als Ganzzahl darstellbare Jahr angegeben werden.

BIS Das letzte Jahr, in dem diese Regel gültig ist. Die Syntax ist identisch mit der des VON-Feldes.

TYP
Gibt den Typ des Jahres an, auf den diese Regel angewendet werden soll. Das Jokerzeichen - bezeichnet alle Jahre. Der Typ des Jahres wird mit dem mit der -y-Option angegebenen Befehl oder mit **yearistype** *Jahr Typ* überprüft. Ein Rückgabewert von 0 bedeutet, daß das Jahr vom angegebenen Typ ist, eine 1, daß es dies nicht ist (siehe -y-Option).

IM Der Monat, in dem diese Regel angewendet werden soll.

AM Der Tag, an dem diese Regel angewendet werden soll. Leerzeichen sind nicht zulässig. Beispiele:

1	Der Erste eines Monats
firstSun	Der erste Sonntag
Sun>=3	Der erste Sonntag nach dem oder am 3.

UM Gibt an, ab welchem Zeitpunkt die Regel gelten soll. Um ein Uhr mittags anzugeben, können Sie 13, 13:00 oder 13:00:00 schreiben. Außerdem kann ein Suffix (ohne Leerzeichen dazwischen) aus folgender Liste angehängt werden:

s	Lokale Standardzeit.
u, g, z	Universalzeit (UTC).
w	Normale Uhrzeit (Default).

SOMMER
Dieser Betrag wird zu der lokalen Standardzeit hinzuaddiert. Das Format ist wie bei UM, aber ohne Suffixe.

BUCHSTABEN
Gibt die Buchstaben an, die in Zeitzonen auf diese Regeln hinweisen sollen (z.B. M für MEZ). Wenn keine Abkürzung angegeben werden soll, verwenden Sie einen -.

Felder in Zonenzeilen

`Zone` *`NAME GMTABST REGELN/SOMMER FORMAT`* `[`*`BIS`*`]`

NAME
Name der Zeitzone.

→

GMTABST
: Abstand zur Greenwich Mean Time (GMT). Wird wie UM formatiert.

REGELN/SOMMER
: Entweder der Name einer Regel, die auf diese Zone angewendet werden soll, oder der Betrag, der zu der lokalen Standardzeit hinzuaddiert werden soll. Damit die Zone identisch mit der lokalen Standardzeit ist, verwenden Sie einen -.

FORMAT
: Gibt an, wie Zeitzonenabkürzungen formatiert werden sollen. Der variable Anteil wird mit %s angegeben.

BIS An diesem Datum soll die Regel für diese Zone gewechselt werden. Die nächste Zeile muß die neue Zoneninformation angeben und darf daher weder den String „Zone" noch das NAME-Feld enthalten.

Felder der Linkzeile

```
Link QUELL-ZEILE ZIEL-ZEILE
```

QUELL-ZEILE
: Der Name der Zone, die gelinkt wird.

ZIEL-ZEILE
: Ein alternativer Name für die in QUELL-ZEILE angegebene Zone.

Felder der Schaltzeile

```
Leap JAHR MONAT TAG HH:MM:SS KORR R/S
```

JAHR MONAT TAG HH:MM:SS
: Gibt an, wann die Schaltsekunde verwendet werden soll.

KORR
: Ein „+" oder „-", das anzeigt, ob die Schaltsekunde hinzugezählt oder abgezogen werden soll.

R/S Eine Abkürzung für *"Rolling"* oder *"Stationary"*, die angibt, ob die Schaltsekunde die lokale Uhrzeit oder GMT beeinflußt.

Index

&-Befehl (ex), 253
< >-Befehl (ex), 252
* bash Shell-Platzhalterzeichen, 129
\ (Backslash), 209
! (Ausrufungszeichen)
    bash Shell-Platzhalterzeichen, 129
    ex-Befehl, 252
[ ] (eckige Klammern)
    bash Shell-Platzhalterzeichen, 129
    Metazeichen in Mustervergleichen, 209
^ Metazeichen in Mustervergleichen, 209
: (Doppelpunkt)
    bash Shell-Befehl, 143
    csh/tcsh-Befehl, 186
    sed-Befehl, 258
$ Metazeichen in Mustervergleichen, 209
. (Punkt)
    bash Shell-Befehl, 143
    Metazeichen in Mustervergleichen, 209
= (Gleichheitszeichen)
    ex-Befehl, 252
    sed-Befehl, 259
    Zuweisung in gawk, 272
(Doppelkreuz), 143, 186
#!-Befehl, 143, 186
– Metazeichen in Mustervergleichen, 209
| Metazeichen in Mustervergleichen, 210
+ Metazeichen in Mustervergleichen, 210
? (Fragezeichen)
    bash Shell-Platzhalterzeichen, 129
    Metazeichen in Mustervergleichen, 210
~-Befehl (ex), 253

A

a-Befehl (sed), 259
Abbrechen
    Emacs-Befehle zum, 216
abbrev-Befehl (ex), 245
Abhängigkeitendatei, 371
Abkürzungen, spezielle tcsh-, 204
Abkürzungsbefehle, Emacs, 217
Ablegen von Verzeichnissen, 145
Abschnitte
    Online-Handbücher, 79
    von Dateien, 21
Absätze, Emacs und, 216, 218
addsuffix-Variable (csh/tcsh), 171
adduser-Befehl, 361
Adressen
    Email, verifizieren, 116
    IP-, 353
    Muster, sed-Editor, 257

Adressen (Forts.)
Zeilen-, ex Editor, 244
agetty-Befehl, 361
aktuelle
Uhrzeit und Datum, 23
alias-Befehl, 143, 186
Aliase, spezielle tcsh-, 204
alloc-Befehl (csh/tcsh), 187
Anmelden, 393, 416
Anpassung der eigenen Arbeitsumgebung, 124
Anpingen von Rechnern, 408
Anweisungen, cpp, 315
append-Befehl (ex), 245
ar-Befehl (Linux), 310
arch-Befehl, 2
Archivdateien, 310, 347, 431
auf Festplatten oder Disketten kopieren, 366-369
Archivierung von Dateien, 18-21
args-Befehl (ex), 245
argv-Variable (csh/tcsh), 171
Arithmetik (siehe Zahlen)
arp-Befehl (TCP/IP), 362
as-Befehl (Linux), 311
atan2-Befehl (gawk), 273
at-Befehl, 2
atq-Befehl, 3
atrm-Befehl, 4
Attribute, Datei, 10, 73
Aufträge
Drucken (siehe Drucken)
Ausdrücke auswerten, 33
Ausführungsrecht (siehe Zugriffsrechte, Datei)
Ausgabe
auf die Standardausgabe, 9
Dateien nach STDOUT, 102
Durchblättern, 61-67, 81-82
echo-Befehl, 146
Terminal I/O-Optionen, 96-100
Umformen Stdin nach Stdout, 112
Umleitung (siehe Umleitung)
Verzweigen, 103
Aushängen von Dateisystemen, 439
Auslagerungsbereich, 398, 428
Ausrichtung, Emacs-Befehle zur, 219
Ausrufungszeichen (siehe ! (Ausrufungszeichen))
Ausschneiden (siehe Kopieren und Einfügen)
autocorrect-Variable (csh/tcsh), 171
autoexpand-Variable (csh/tcsh), 171
autolist-Variable (csh/tcsh), 171
autologout-Variable (csh/tcsh), 171
awk-Befehl, 210
awk-Sprache, 208

B

b-Befehl (sed), 259
backslash_quote-Variable (csh/tcsh), 171
Backups (siehe Archivierung von Dateien)
badblocks-Befehl, 363
banner-Befehl, 4
bash Shell, 127-161
Eingebaute Befehle, 143-160
bash-Befehl, 4
batch-Befehl, 4
bc Programmiersprache, 5-9
Beenden
des ex-Editors, 243
vi-Befehle zum, 235
Befehle
Aliase für, 143
bash Shell, 135, 143-160
csh und tcsh, 167-168, 186-204
Emacs, 214-227
ex-Editor, 243-253
FTP, 45-50
gawk-Sprache, 273-278
gdb (GNU Debugger), 326
ipfwadm, 389
Linux, 309-345
Mail, 74-77
NIS, 441-445
nslookup-Werkzeug, 403
Problemsuche mit TCP/IP, 357
sed-Editor, 256-265
Shell-, 125-126
Suchen nach, 149
Systemverwaltung, 347-351, 361-448
TCP/IP-Verwaltung, 352
Telnet, 104
TFTP, 111
top-Befehl, 112

Befehle (Forts.)
Vervollständigung (siehe Vervollständigung)
vi-Editor, 230-237
Befehls-Ersetzung (csh/tcsh), 179
Befehls-History (siehe History, Befehls-)
Befehlsmodus, vi, 230
Benachrichtigung bei ankommender Mail, 9
Benutzer
Anlegen, 361
Ansprechen, 431
Befehle zur Verwaltung, 351
bei ankommender Mail benachrichtigen, 9
auf einem bestimmten Rechner, Anzeigen, 421
Dialog mit, 103
Einloggen, 69, 89
finger-Befehl, 41
Gruppenzugehörigkeit ändern, 83
Identifikationsnummer, 118
Informationen über, 57, 118
Liste, 92, 115
Meldungen schicken, 421
NIS-Zugänge, 360
Schreiben an, 93, 118, 440
su-Befehl, 100
Benutzernamen, 69
Berkeley Software Distribution (BSD), x-xi
Beschreibungsdatei (siehe make-Werkzeug)
Bewerten von Ausdrücken, 109
bg-Befehl, 144, 187
Bibliotheken
Anzeigen des Inhalts von, 340
Erzeugen von, 310
Linken mit, 332
biff-Befehl, 9
/bin/echo-Befehl (siehe echo-Befehl)
BIND (Berkeley Internet Name Domain), 356
bind-Befehl, 144
bindkey-Befehl (tcsh), 188
bison-Befehl (Linux), 311
Blöcke, Suchen nach schlechten, 363
bootpd-Befehl (TCP/IP), 363
bootpgw-Befehl, 364
bootptest-Befehl (TCP/IP), 365
break-Befehl, 144, 188, 273
breaksw-Befehl (csh/tcsh), 188
BSD (Berkeley Software Distribution), x-xi
Buffer
auf Festplatte schreiben, 429
Emacs-Befehle für, 217
built-in Befehl (bash), 144
built-ins-Befehl (tcsh), 188
bye-Befehl (tcsh), 188

C

c-Befehl (sed), 259
C Programmiersprache
cpp Präprozessor, 313-316
Debuggen, 325-327
gcc-Compiler, 319-325
c++-Befehl (siehe g++-Befehl)
C++-Programmiersprache, 319
Debuggen, 325-327
cal-Befehl, 9
case-Befehl (csh/tcsh), 188
cat-Befehl, 9
C-Befehle, Emacs, 220-222
C-s Befehl (Emacs), 214
cc-Befehl (siehe gcc-Compiler)
cd-Befehl, 145, 188
cfdisk-Befehl, 365
change-Befehl (ex), 245
chattr-Befehl, 10
chdir-Befehl (csh/tcsh), 189
chfn-Befehl (NFS/NIS), 11
chgrp-Befehl, 11
chmod-Befehl, 12
chown-Befehl, 13
chroot-Befehl, 366
chsh-Befehl, 14
Circumflex (^) Metazeichen, 209
cksum-Befehl, 14
clear-Befehl, 14
clock-Befehl, 366
close-Befehl (gawk), 273
cmp-Befehl, 14
colcrt-Befehl, 15
colrm-Befehl, 15
column-Befehl, 15
command-Befehl, 145
comm-Befehl, 16
complete-Befehl (tcsh), 189
complete-Variable (csh/tcsh), 172
compress-Befehl, 53

continue-Befehl, 145, 190, 273
copy-Befehl (ex), 245
copyleft (siehe GPL)
correct-Variable (csh/tcsh), 172
cos-Befehl (gawk), 273
cp-Befehl, 16
cpio-Befehl, 18-21, 366-369
cpp Präprozessor, 313-316
  imake-Schnittstelle, 328-331
CPU Last (siehe Performance)
CRC (cyclic redundancy check), 14
cron-Befehl, 369
csh Shell, 21, 163-205
  beispielhafte .cshrc-Datei, 176
csplit-Befehl, 21
ctags-Befehl (Linux), 316
Cursor-Bewegung
  Emacs-Befehle zur, 215
  vi-Befehle zur, 232
cut-Befehl, 23
cwd-Variable (csh/tcsh), 172
Cyclic Redundancy Check (CRC), 14

D

d-Befehl (sed), 260
D-Befehl (sed), 260
Darstellung der mittleren Systemlast, 111
date-Befehl, 23
Dateien
  Abhängigkeiten-, 371
  Archiv- (siehe Archivdateien)
  Attribute, 10, 73
  auf mehreren Rechnern verwalten, 412-414
  auflisten von, 71-73
  Aufteilen, 96
  Ausgeben, 9, 102
  für bash Shell, 129
  benutzende Prozesse identifizieren, 376
  Blockieren, 68
  csh/tcsh, 165
  doppelte Textzeilen löschen, 114
  Drucken, 54, 69-71, 102
  Durchblättern, 61-67, 81-82
  Eigentümer, 13
  Elemente zählen, 117
  Emacs-Befehle für, 215
  imake-Konfiguration, 330
  Klassifizieren nach Typ, 36
  Kompression von, 53, 121
  Kopieren, 16, 89, 93, 366-369
  Löschen, 90
  Mail-bezogen, 78
  Objekt- (siehe Objektdateien)
  Pseudonyme für (siehe symbolische Links)
  Seitenformatierung, 85
  Sortierung des Inhalts von, 95
  Spalten zusammenführen von, 83
  Suchen nach, 37-41, 68
  Suchen nach Inhalten von, 31, 36, 51, 69, 121
  Testen, 109
  Überprüfen, 14, 84
  Übertragen, 45-50, 110
  Umbenennen, 82
  Unterteilung, 21
  Vergleichen, 14, 16, 26-29, 121
  vi-Befehle für, 235
  Zeitkonvertierung, 446
  Zugriffsrechte (siehe Zugriffsrechte, Datei)
Dateinamen, Mustervergleiche, 165, 207
Dateisysteme
  Aushängen, 439
  Befehle zur Verwaltung von, 349
  benutzende Prozesse identifizieren, 376
  Buffer auf Festplatten schreiben, 429
  Einhängen, 399-401
  Erzeugen, 397
  MINIX, 375, 398
  Neustarten, 415
  NFS, 358-360
  Second Extended, 370, 372, 397, 437
Datum (siehe Uhrzeit und Datum)
dbm Dateien, 395
debugfs-Befehl, 370
Debuggen
  ext2-Dateisystem, 370
  gdb (GNU Debugger), 325-327
declare-Befehl, 145
default-Befehl (csh/tcsh), 190
defekte Dateien, 14
Deklarieren von Variablen und Feldern (gawk), 272

Dekompression (siehe Kompression, Datei)
delete-Befehl, 246, 273
depmod-Befehl, 371
df-Befehl, 25
diff3-Befehl, 28
diff-Befehl, 26
dirs-Befehl, 145, 190
dirstack-Variable (csh/tcsh), 172
Disketten
  Formatieren, 35
  Kopieren von Archiven auf, 366-369
  Parameter, 93
Display (siehe Terminals)
dmesg-Befehl, 372
DNS (Domain Name Service), 356
dnsdomainname-Befehl (TCP/IP), 372
do-Befehl (gawk), 274
Dokumentation, xii-xiii, 78
  Emacs, 219
  Infodateien, 58
  Manualpages, 1, 78
  TCP/IP, 353
domainname-Befehl (NFS/NIS), 372
Domänen, 356, 359
Drucken, 69-71, 350
  Dateien auf die Standardausgabe, 9, 102
  Einstellen, 438
  lpd-Dämon, 394
  Spruchband, Ausgabe als, 4
du-Befehl, 29
dumpe2fs-Befehl, 372
dumpkeys-Befehl, 30
dunique-Variable (csh/tcsh), 172
Durchsuchen
  Dateiinhalt, 31, 36, 51
  Manualpages, 1, 117
Dämonen
  Befehle für, 347
  NFS-, 358
  Routing, 355

E

e2fsck-Befehl, 372
echo-Befehl, 30, 146, 190
echo_style-Variable (csh/tcsh), 172
echotc-Befehl (csh/tcsh), 190
echo-Variable (csh/tcsh), 172
eckige Klammern (siehe [ ])
ed-Editor, 208, 210
edit-Befehl (ex), 246
Editieren
  auf der Kommandozeile, 182-186, 257
  vi-Befehle zum, 233
Editiermodus (bash), 141
egrep-Befehl, 31, 210
  Metazeichen im, 208
Eigentümer
  Datei, 13
Ein-/Ausgabebefehle, sed, 258
Einfügemodus, vi, 230
Eingabe
  in Mailbox-Format filtern, 43-45
  Kommandos von der Standardeingabe, 4, 119
  Terminal I/O-Optionen, 96-100
  Umformen Stdin nach Stdout, 112
  Umleitung (siehe Umleitung)
Eingabeaufforderung, Format (tcsh), 175
Einhängen von Dateisystemen, 358, 399-401
  Aushängen, 439
Einrückungen, Emacs-Befehle für, 218
electronic mail (siehe Mail)
ellipsis-Variable (csh/tcsh), 172
Emacs Editor, 32, 210, 213-227
Emacs-artige Befehle, 130-131
Emacs-Modus (csh/tcsh), 183
Email (siehe Mail)
Entfernen (siehe Löschen)
Entfernte Rechner (siehe Rechner)
Entf-Taste, Emacs und die, 213
Entladen von Modulen, 417
env-Befehl, 32
Ersetzen, Metazeichen beim, 211-212
Ersetzungsoperatoren
  csh/tcsh-Shells, 169
etags-Befehl (Linux), 317
/etc/groups-Datei, 11
/etc/magic-Datei, 36
/etc/passwd-Datei, 124, 361
  Ändern von Informationen in, 11, 120
eval-Befehl (csh/tcsh), 191
exec-Befehl, 146, 191
ex-Editor, 33, 210, 243-253
  Metazeichen im, 208
exit-Befehl, 146, 192, 274

expand-Befehl, 33
exp-Befehl (gawk), 274
export-Befehl, 147
Exportieren von Dateisystemen, 358
expr-Befehl, 33
ext2-Dateisystem, 370, 372, 397, 437

F

false-Befehl, 35
fc-Befehl, 142, 147
fdformat-Befehl, 35
fdisk-Befehl, 373
Felder, in gawk, 272
Fenster, Emacs-Befehle für, 218
Festplatten
  Benutzungs-Informationen über, 411
  Buffer von Dateisystemen schreiben, 429
  Kopieren von Archiven auf, 366-369
  Partitionieren, 365, 373
fg-Befehl, 148, 192
fgrep-Befehl, 36
File Transfer Protocol (FTP), 45-50
file-Befehl, 36, 246
filetest-Befehl (tcsh), 192
find-Befehl, 37-41
Finden (siehe Suchen)
finger-Befehl, 11, 41, 120
Firewalls, 387
flex-Befehl (Linux), 318
fmt-Befehl, 42
fold-Befehl, 42
for-Befehl (gawk), 274
foreach-Befehl (csh/tcsh), 192
formail-Befehl, 43-45
Format
  Mailbox-, Standardeingabe filtern, 43-45
  tcsh-Eingabeaufforderung, 175
  Uhrzeit und Datum, 23
Formatieren
  Dateien als Seiten, 85
  Disketten, 35
for-Schleife, 148
Fragezeichen (siehe ? (Fragezeichen))
Free Software Foundation (FSF), xi
free-Befehl, 45
fsck.minix-Befehl, 375
ftp-Befehl, 45-50
function-Befehl (gawk), 274
Funktionen
  bc Programmiersprache, 7
  ctags- und etags-Befehle (Linux), 316-318
  Funktionsbindungen, 144
  make-Werkzeug, 338
fuser-Befehl, 51, 376

G

G-Befehl (sed), 260
g-Befehl (sed), 260
g++-Befehl (Linux), 319
gated-Dämon, 355, 376
Gateways, 355
gawk-Skriptsprache, 51, 267-278
gcc-Compiler, 319-325
gdb Debugger, 325-327
gdc-Befehl (TCP/IP), 377
General Public License (GPL), xi
Geschichte von Linux, ix
getkeycodes-Befehl, 51
getline-Befehl (gawk), 274
getopts-Befehl, 148
getty-Befehl, 379
gid-Variable (csh/tcsh), 172
global-Befehl (ex), 246
glob-Befehl (csh/tcsh), 193
GNU C Präprozessor, 313-316
GNU-Projekt, x
goto-Befehl (csh/tcsh), 193
GPL (General Public License), xi
gprof-Befehl (Linux), 328
grep-Befehl, 51, 210
  Metazeichen im, 208
Groß-/Kleinschreibung, Emacs-Befehle zur, 217
Gruppen, 11, 83
gsub-Befehl (gawk), 275
gzexe-Befehl, 53
gzip-Befehl, 53

H

H-Befehl (sed), 261
h-Befehl (sed), 261
halt-Befehl, 380
HANGUP, 83
Hardware, Befehle zur Kontrolle von, 348
hash-Befehl, 149

hashstat-Befehl (csh/tcsh), 193
head-Befehl, 54
Hilfe, Emacs-Befehle zur, 219-220
Hintergrund-Jobs, 144, 205
histdup-Variable (csh/tcsh), 172
histfile-Variable (csh/tcsh), 172
histlit-Variable (csh/tcsh), 172
History, Befehls-
    bash Shell, 130, 133, 139
    csh und tcsh, 179-181
    fc-Befehl, 147
history-Befehl (csh/tcsh), 193
history-Variable (csh/tcsh), 172
home-Variable (csh/tcsh), 172
host-Befehl, 55-57, 380
hostname-Befehl, 57
hup-Befehl (tcsh), 194

I

i-Befehl (sed), 261
ibase Schlüsselwort, 5
icmpinfo-Befehl (TCP/IP), 382
id-Befehl, 57
if-Befehl, 150, 194, 275
ifconfig-Befehl, 357, 383
ignoreeof-Variable (csh/tcsh), 172
ignore-Variable (csh/tcsh), 172
Imakefile Datei, 331
imake-Schnittstelle (Linux), 328-331
index-Befehl (gawk), 275
inetd-Dämon, 385
info-Befehl, 58
Inhalte
    Datei (siehe Dateien)
init-Befehl, 386
inkrementelle Suchbefehle, 217
insert-Befehl (ex), 246
insmod-Befehl, 387
install-Befehl, 387
int-Befehl (gawk), 275
Integrität, System-, 351
interaktive Shell-Benutzung, 123
Internet Protocol (siehe TCP/IP)
IP (siehe TCP/IP)
IP-Adressen, 353
ipfwadm-Befehle (TCP/IP), 389
ipfw-Befehl (TCP/IP), 387
ispell-Befehl, 59

J

Jobs, 3
    bash Shell und, 160-161
    Hintergrund/Vordergrund, 144, 148, 205
    Verwaltung, 205
jobs-Befehl (csh/tcsh), 195
join-Befehl (ex), 247

K

k-Befehl (ex), 247
Kalender, 9
kbd_mode-Befehl, 60
kbdrate-Befehl, 390
Kernel, ix
    Ausgeben der exportierten Symbole, 392
    Befehle zur Verwaltung, 349
    Einstellen des Videomodus, 440
keyboard
    dumpkeys-Befehl, 30
kill-Befehl, 60, 195
killall5-Befehl, 61
killall-Befehl, 60
Klassifizieren
    Dateien nach Typ, 36
klogd-Befehl, 391
Kommandos
    Ausführen auf einem entfernten Rechner, 91
    Ausführen von der Standardeingabe, 4
    Beenden nach HANGUP, 83
    xargs-Befehl, 119
    zmore-Befehl, 121
Kommandozeile
    Editieren
        bash-Befehle, 130-133
        csh/tcsh-Befehle, 182-186
    gawk-Sprachsyntax, 268
    sed-Editor-Syntax, 256
    Verarbeiten der Parameter, 148
    vi-Editor Syntax, 230
Kommunikation über serielle Verbindungen, 357, 428
Kompilieren von C-Quelldateien, 319-325
Kompression, Datei, 53, 121
    Ändern .Z-Dateien nach .gz, 122
Konfigurieren von TCP/IP, 357

Konvertieren von .Z-Dateien nach .gz, 122
Kopieren und Einfügen
  sed, 258
Kopieren und Verschieben
  vi, 234
Kopieren von Dateien, 16, 89, 387
  Datei-Archive, 18-21, 366-369
Koprozesse, 137
ksyms-Befehl, 392

L

l-Befehl (sed), 262
Laden von Modulen, 387, 399
  Entladen, 417
last Schlüsselwort, 5
ld (Linker), 331
ldconfig-Befehl, 392
ldd-Befehl (Linux), 333
length-Befehl (gawk), 275
Leserecht (siehe Zugriffsrechte, Datei)
less-Befehl, 61-67
limit-Befehl (csh/tcsh), 196
Linux
  Geschichte, ix
  Quelltext, xi
list-Befehl (ex), 247
listjobs-Variable (csh/tcsh), 173
listlinks-Variable (csh/tcsh), 173
listmaxrows-Variable (csh/tcsh), 173
listmax-Variable (csh/tcsh), 173
Lizenzen, xi
ln-Befehl, 67
locate-Befehl, 68
lockfile-Befehl, 68
log-Befehl, 196, 275
logger-Befehl (TCP/IP), 393
login-Befehl, 69, 197, 393
Loginshell (siehe Shells)
logische Operatoren, 34
logname-Befehl, 69
logut-Befehl (csh/tcsh), 197
look-Befehl, 69
lpd-Dämon, 394
lpq-Befehl, 69
lpr-Befehl, 70
lprm-Befehl, 71
lsattr-Befehl, 73
ls-Befehl, 71-73
ls-F-Befehl (tcsh), 197
lsmod-Befehl, 395
Löschen
  Dateien, 90
  doppelte Textzeilen, 114
  Druckaufträge von der Warteschlange, 71
  Emacs-Befehle zum, 216
  Spalten, 15, 23
  Verzeichnisse, 90
  vi-Befehle zum, 234
  wartende Jobs, 4

M

M-Befehle, Emacs, 222-223
m4-Prozessor, 333
magic-Datei, 36
Mail, 74-78
  Adressen verifizieren, 116
  bash Shell-Variablen, 140
  Befehle zur Verwaltung von, 348
  Benutzer bei ankommender Mail benachrichtigen, 9
  Post Office Protocol, 85
  Standardeingabe in Mailbox-Format filtern, 43-45
.mailrc-Datei, 77
mail-Variable (csh/tcsh), 173
makedbm-Befehl, 395
makedepend-Werkzeug (imake), 329
makemap-Befehl, 397
make-Werkzeug (Linux), 334-340
  imake-Schnittstelle, 328-331
Makros
  ctags- und etags-Befehle (Linux), 316-318
  Emacs-Befehle, 219
  m4-Prozessor, 333
  make-Werkzeug, 337
  vi-Befehle für, 236
man-Befehl, 78
manpath-Befehl, 79
Manualpages
  Durchsuchen, 1, 117
map-Befehl (ex), 247
mark-Befehl (ex), 247
Maschine
  Architektur, 2
  Laufzeit, 420
matchbeep-Variable (csh/tcsh), 173
match-Befehl (gawk), 276

Meldungen
an alle Benutzer schicken, 421, 440
System-, 372, 391
mesg-Befehl, 80
Metazeichen, 208-210
Expansion von Dateinamen, 207
Suchen von Dateinamen, 165
MINIX-Dateisysteme, 375, 398
mkdir-Befehl, 80
mkdirhier-Werkzeug (imake), 330
mke2fs-Befehl, 397
mkfs-Befehl, 398
mkfs.minix-Befehl, 398
mklost+found-Befehl, 398
mkswap-Befehl, 398
modprobe-Befehl, 399
Module
Anzeigen, 395
Entladen, 417
Laden, 387, 399
more-Befehl, 81-82
mount-Befehl, 399-401
move-Befehl (ex), 247
Musteradressierung (sed), 257
Mustervergleiche, 207-212
bash-Operatoren für, 138
in der gawk-Sprache, 269-271
mv-Befehl, 82

N

n-Befehl (sed), 262
N-Befehl (sed), 262
Nachrichten
an Terminals, 80
Name Service, 356
named-Dämon, 401
Namen
Benutzernamen, 69
Dateien und Verzeichnisse, 82
Domänennamen, 356
von Emacs-Befehlen, 223-227
Hostnamen, 57
Pseudonyme für Dateien erstellen, 67
netdate-Befehl (TCP/IP), 402
netstat-Befehl (TCP/IP), 402
Netzwerke, 351-353
Abfragen des Zustands, 402, 430
Befehle zur Verwaltung, 350
TCP/IP-, 353-357
newgrp-Befehl, 83, 197
next-Befehl, 247, 276
NFS (Network File System), 358-360
nice-Befehl, 83, 197
NIS (Network Information Service), 359-360
Befehle, 441-445
Befehle zur Verwaltung, 350
nm-Befehl (Linux), 340
noclobber-Variable (csh/tcsh), 173
noglob-Variable (csh/tcsh), 173
nohup-Befehl, 83, 197
nonomatch-Variable (csh/tcsh), 173
notify-Befehl (csh/tcsh), 197
nslookup-Werkzeug (TCP/IP), 403-405
number-Befehl (ex), 248
Nummern
Zeilennummern/-adressen, 233, 244

O

obase Schlüsselwort, 5
Objektdateien
Erzeugen von, 311
Größe von, 344
Profiling-Daten, 328
Symboltabellen von, 340
Oktalzahlen für Zugriffsrechte, 13
onintr-Befehl (csh/tcsh), 198
Online Dokumentation (siehe Dokumentation)
open-Befehl (ex), 248
Operatoren, 33
bash Shell, 137-138, 140
bc Programmiersprache, 8
csh und tcsh, 177-179
gawk-Skriptsprache, 272
logische, 34
owd-Variable (csh/tcsh), 173

P

p-Befehl (sed), 262
P-Befehl (sed), 263
Partitionieren von Festplatten, 365, 373
paste-Befehl, 83
patch-Befehl (Linux), 341
pathchk-Befehl, 84
Paßwörter, 120, 442

Performance
CPU Last, 111
Darstellung der mittleren Systemlast, 111
Einstellen der Tastaturwiederholrate, 390
perl Programmiersprache, 84, 210
Pfad
aktuelles Arbeitsverzeichnis, 88
Online Handbuch, 79
pidof-Befehl, 84
PIDs (Prozeßidentifikationsnummern), 51, 84
ping-Befehl, 408
.plan-Datei, 41
Platten
Platz auf (siehe Speicher)
popclient-Befehl, 85
popd-Befehl (csh/tcsh), 198
Post Office Protocol (POP), 85
powerd-Dämon, 409
PPP (Point-to-Point Protocol), 357, 409
pppd-Dämon, 409
pr-Befehl, 85
preserve-Befehl (ex), 248
print-Befehl, 248, 276
printenv-Befehl (csh/tcsh), 198
printexitvalue-Variable (csh/tcsh), 173
printf-Befehl (gawk), 276
PRIO_MAX, PRIO_MIN Umgebungsvariable, 89
Priorität
Meldung, 391
Prozeß, 83, 89
Problembeseitigung
Emacs Editor, 213
Problemsuche
TCP/IP, 357
Profiling-Daten, Objektdatei, 328
Programme
Debuggen (siehe Debuggen)
Linux, x
Shell-Skripten, 124
.project-Datei, 41
prompt2, prompt3-Variablen (csh/tcsh), 173
Protokolle, TCP/IP, 353, 355
Prozeduren, in der gawk-Sprache, 270-271
Prozesse, 86
Abbrechen, 60-61
Anhalten, 380
Ansehen der CPU-Intensiven, 111
Befehle zur Verwaltung von Prozessen, 351
exec-Befehl, 146
Identifikationsnummern (PIDs), 51, 84
Koprozesse (bash), 137
nach Datei oder Dateisystem identifizieren, 376
Priorität, 83, 89
ps-Befehl, 86
Pseudonyme für Dateien (siehe symbolische Links)
psupdate-Befehl, 411
pushd-Befehl (csh/tcsh), 198
pushdsilent-Variable (csh/tcsh), 174
pushdtohome-Variable (csh/tcsh), 174
put-Befehl (ex), 248
pwd-Befehl, 88

Q

q-Befehl (sed), 263
Quelltext, Linux, xi
quit-Befehl (ex), 248
Quoten, 134, 166

R

r-Befehl (sed), 263
ramsize-Befehl, 411
rand-Befehl (gawk), 277
rarp-Befehl, 411
rcp-Befehl, 89
rdate-Befehl (TCP/IP), 411
rdev-Befehl, 412
rdist-Befehl, 412-414
rdistd-Dämon, 414
read-Befehl (ex), 249
reboot-Befehl, 415
recexact-Variable (csh/tcsh), 174
Rechner
angemeldete Benutzer ausgeben, 421
Anpingen, 408
auf entfernte Rechner zugreifen (siehe Telnet)
auf entfernten Rechnern anmelden, 416
Befehle zur Verwaltung, 348

Rechner (Forts.)
Dateien auf mehreren Rechnern verwalten, 412-414
entfernt Einloggen, 89
Informationen über, 55-57, 92, 420, 430
Meldungen an Benutzer schicken, 421
Paketrouten verfolgen, 437
Programmausführung auf entfernten, 91
Schreiben an alle Benutzer, 93
Status, 92
Rechnerarchitektur, 2
Rechte
Zugriff auf eine Datei (siehe Zugriffsrechte, Datei)
Rechtschreibprüfung, 59
recognize_only_executables-Variable (csh/tcsh), 174
recover-Befehl (ex), 249
Regionen, im Emacs, 216
Registrieren von Domänen, 356
Reguläre Ausdrücke (siehe Mustervergleiche)
Durchsuchen Dateiinhalt, 31, 51
rehash-Befehl (csh/tcsh), 199
renice-Befehl, 89
repeat-Befehl (csh/tcsh), 199
reset-Befehl, 89
Ressourcen (siehe Dokumentation)
return-Befehl (gawk), 277
Reverse Address Resolution Protocol (RARP), 411
rewind-Befehl (ex), 249
rexecd-Dämon, 415-416
rlogin-Befehl, 89
rlogind-Dämon, 416
rmail-Befehl (TCP/IP), 417
rm-Befehl, 90
rmdir-Befehl, 90
rmmod-Befehl, 417
rootflags-Befehl, 417
route-Befehl (TCP/IP), 417
routed-Dämon, 355, 417-418
Routing, 355, 417-418
RPC (Remote Procedure Call), 360, 419
rpcgen-Befehl (Linux), 343
rpcinfo-Befehl (NFS/NIS), 91, 419
rsh-Befehl, 91
rshd-Dämon, 420
rstat-Befehl, 92, 420
Runlevel, System, 420, 435
run-parts-Befehl, 92, 420
ruptime-Befehl, 92, 420
rusers-Befehl, 92, 421
rwall-Befehl, 93, 421
Rückgängigmachen, Emacs-Befehle zum, 216
Rückschritt-Taste, Emacs und die, 213

S

s-Befehl (sed), 263
savehist-Variable (csh/tcsh), 174
scale Schlüsselwort, 5
sched-Befehl (csh/tcsh), 199
sched-Variable (csh/tcsh), 174
Schleifen, 144, 148
Schreiben (siehe Meldungen)
auf Festplatte, 429
an Benutzer, 93, 118, 440
an Terminal, 80
Schreibrecht (siehe Zugriffsrechte, Datei)
schwebende Jobs (siehe Jobs)
script-Befehl, 93, 421-427
Second Extended Dateisystem, 370, 372, 397, 437
sed-Editor, 93, 210, 255-265
Metazeichen im, 208
sendmail, makemap-Befehl und, 397
Serielle Verbindungen
Kommunikation, 428
Server
Informationen über, 91, 94
NFS, 358-360
NIS, 359-360
rshd-Dämon, 420
wegen Datums- oder Uhrzeitänderung benachrichtigen, 411
:set-Befehl (vi), 239-242
set-Befehl, 200, 249
setdprm-Befehl, 93
setenv-Befehl (csh/tcsh), 200
settc-Befehl (csh/tcsh), 200
setty-Befehl (tcsh), 200
sh Shell, 94
shell-Befehl (ex), 249
Shells, xiii, 123-126

Shells (Forts.)
Emacs-Zeichen im Shell-Modus, 218
Loginshell ändern, 120
mit bestimmter EUID erzeugen, 100
rshd-Dämon, 420
Shell-Skripten, 124
vi-Befehle für, 236
Ändern der Loginshell, 14
Shell-Variablen
bash Shell, 138-140
csh/tcsh, 174
Exportieren von Werten, 147
shift-Befehl (csh/tcsh), 201
showdots-Variable (csh/tcsh), 174
showmount-Befehl (NFS/NIS), 94
shutdown-Befehl, 427
Sicherheit, Befehle zur Verwaltung der System-, 351
Signaldateien, 68
sin-Befehl (gawk), 277
size-Befehl (Linux), 344
Skripten, 92
alle in einem Verzeichnis ausführen, 420
Beenden, 146
gawk- (siehe gawk-Programmiersprache)
slattach-Befehl (TCP/IP), 428
sleep-Befehl, 94
SLIP (Serial Line IP), 357
Sonderzeichen
bash Shell, 134
bc Programmiersprache, print Erweiterung, 6
colcrt-Befehl, 15
tr-Befehl, 112
sort-Befehl, 95
Sortierung von Dateiinhalten, 95
source-Befehl, 201, 249
Spalten
Formatieren der Eingabe in, 15
Löschen, 15, 23
Zusammenführen von Text in, 83
Speicher, 45
freier Plattenplatz, 25
Plattenbelegung, 29
Speichern, vi-Befehle zum, 235
split-Befehl (gawk), 277
sprintf-Befehl (gawk), 277
sqrt-Befehl (gawk), 277
srand-Befehl (gawk), 277
Standardausgabe (siehe Ausgabe)
Standardeingabe (siehe Eingabe)
status-Variable (csh/tcsh), 174
stop-Befehl (csh/tcsh), 201
strftime-Befehl (gawk), 278
strip-Befehl (Linux), 344
Stromversorgung
unterbrechungsfreie, 409
stty-Befehl, 96-100
sub-Befehl (gawk), 278
su-Befehl, 100
substitute-Befehl (ex), 250
Substitutions-Operatoren
bash Shell, 137, 142
substr-Befehl (gawk), 278
Suchen
Dateiinhalte, 69, 121
Emacs-Befehle zum, 217
und Ersetzen, 211-212
Metazeichen beim, 209-212
nach Befehlen, 149
nach Dateien, 37-41, 68
nach schlechten Blocks, 363
vi-Befehle zum, 232
suspend-Befehl (csh/tcsh), 202
Swap-Bereich, 398, 428
swapdev-Befehl, 428
swapon, swapoff-Befehle, 429
switch-Befehl (csh/tcsh), 202
Symbole
ctags- und etags-Befehle (Linux), 316-318
symbolische Links, 67, 101
symlinks-Befehl, 101
symlinks-Variable (csh/tcsh), 174
sync-Befehl, 429
syslogd-Dämon, 429
systat-Befehl, 101, 430
System
Aktivität (siehe Performance; Prozesse)
Integrität des (siehe Sicherheit, Befehle zur Verwaltung des Systems)
Kontrollmeldungen, 372
Runlevel, 420, 435
Status, 116, 118, 140
Zeit einstellen, 402
Zusammenfassung ausgeben, 440

System (Forts.)
system-Befehl (gawk), 278
Systemverwaltung, 347-448
Befehle zur, 351
systime-Befehl (gawk), 278

T

t-Befehl (sed), 264
t-Befehl (ex), 250
Tabellen, NIS-, 359, 443
Tabs (siehe Whitespace)
tac-Befehl, 102
tag-Befehl (ex), 250
tail-Befehl, 102
talk-Befehl, 103
talkd-Dämon, 431
tar-Befehl, 431
Tastatur
Modus, 60
Tastaturwiederholrate, 390
Tastaturbindungen, 144
tcpdchk-Befehl (TCP/IP), 435
tcpd-Dämon, 434
tcpdmatch-Befehl (TCP/IP), 435
TCP/IP, 353-357, 434
Befehle zur Verwaltung, 352
tcsh Shell, 103, 163-205
tee-Befehl, 103
Teilen von Dateien, 96
telinit-Befehl, 435
telltc-Befehl (csh/tcsh), 202
telnet-Befehl, 103-109
telnetd-Dämon, 436
Terminals
bash Shell-Variablen für, 139
Bildschirm löschen, 14, 89
getty, agetty-Befehle, 361, 379
I/O-Optionen, 96-100
Schreiben von Nachrichten an, 80
test-Befehl, 109
Text
Kopieren, vi-Befehle zum, 234
Rechtschreibprüfung von, 59
Whitespace (siehe Whitespace)
Zeilen umbrechen, 42
Textdateien (siehe Dateien)
Textzeilen
doppelte entfernen, 114
Zählen, 117
TFTP (Trivial File Transfer Protocol), 110
tftpd-Dämon, 436
time-Befehl (csh/tcsh), 202
time-Variable (csh/tcsh), 174
tload-Befehl, 111
tolower-Befehl (gawk), 278
top-Befehl, 111
toupper-Befehl (gawk), 278
tperiod-Variable (csh/tcsh), 174
traceroute-Befehl (TCP/IP), 437
tr-Befehl, 112
Trivial File Transfer Protocol (TFTP), 110, 436
true-Befehl, 114
tune2fs-Befehl, 437
tunelp-Befehl, 438

U

Übertragen von Dateien, 45-50, 110
Uhrzeit, 347
Uhrzeit und Datum
aktuelle, 23
Dateien zur Zeitkonvertierung, 446
Einstellen, 402
Kalender, 9
Server benachrichtigen, 411
uptime-Befehl, 115
Warten auf, 94
Zeitplan der Programmausführung, 2
Zeitzonen, 445
UID (siehe Benutzer)
ul-Befehl, 114
umask-Befehl (csh/tcsh), 203
Umbenennen Dateien/Verzeichnisse, 82
Umbrechen
Zeilen, 42
Umgebungsvariablen, 32, 176
umgekehrte Ausgabe, 102
Umleitung
csh und tcsh, 167
in der bash Shell, 135-137
umount-Befehl, 439
unabbreviate-Befehl (ex), 250
unalias-Befehl (csh/tcsh), 203
uncomplete-Befehl (tcsh), 203
undo-Befehl (ex), 250
unexpand-Befehl, 114
unhash-Befehl (csh/tcsh), 203

uniq-Befehl, 114
UNIX, x
unlimit-Befehl (csh/tcsh), 203
unmap-Befehl (ex), 250
unset-Befehl (csh/tcsh), 203
unsetenv-Befehl (csh/tcsh), 203
Unterstriche/Unterstreichen, 114
update-Befehl, 439
UPS (Uninterruptible Power Supply), 409
uptime-Befehl, 115
users-Befehl, 115
/usr/lib/magic-Datei, 36
USV (Unterbrechungsfreie Stromversorgung), 409

V

v-Befehl (ex), 251
Variablen
  bash Shell und, 138-140
  csh und tcsh, 168-177
  gawk-Skriptsprache, 271-272
  in der bash Shell, 137-140
  Shell (siehe Shell Variablen)
  Umgebungs- (siehe Umgebungsvariablen)
verbose-Variable (csh/tcsh), 174
Vergleichen von Dateien, 14, 16, 26-29, 121
Verknüpfungen (siehe symbolische Links)
version-Befehl (ex), 251
version-Variable (csh/tcsh), 174
Vertauschen von Text, Emacs-Befehle zum, 216
Vervollständigung
  bash Befehle zur, 131
  csh/tcsh-Befehle zur, 182
Verzeichnisse
  Ablegen, 145
  aktuelle, 88
  alle Skripten darin ausführen, 420
  Anzeige des Inhalts von, 71-73
  Dateien hinein kopieren, 387
  Erstellen, 80
  lost+found, 398
  Löschen, 90
  Umbennen, 82
  Wechseln, 145
  Wurzel-, 366
Verzweigungsbefehle, sed, 258
vi-artige Befehle, 131-133
vidmode-Befehl, 440
vi-Editor, 115, 229-242
  Einstellungen, 239-242
  Metazeichen im, 208
vi-Modus (csh/tcsh), 184
visiblebell-Variable (csh/tcsh), 175
visual-Befehl (ex), 251
Vordergrund-Jobs, 148, 205
vrfy-Befehl, 116

W

w-Befehl, 116, 440
  (sed), 265
wait-Befehl (csh/tcsh), 203
wall-Befehl, 440
Warten (siehe sleep-Programm)
Warteschlange
  Auftrag (siehe Aufträge)
  Druck (siehe Drucken)
watchlog-Befehl (csh/tcsh), 203
watch-Variable (csh/tcsh), 175
wc-Befehl, 117
Weitergabe von Linux, xi
whatis-Befehl, 117
where-Befehl (tcsh), 204
which-Befehl (csh/tcsh), 204
while-Befehl, 204, 278
Whitespace
  colcrt-Befehl, 15
  Emacs-Einrück-Befehle, 218
  fmt-Befehl, 42
  Tabs von/zu Leerzeichen, 33, 114
who-Befehl, 118
whoami-Befehl, 118
Wörter Anzahl, 117
Wörterbuch, Rechtschreibung, 59
wordchars-Variable (csh/tcsh), 175
Wort-Abkürzungsbefehle, Emacs, 217
Wort-Ersetzung (csh/tcsh), 180
wq-Befehl (ex), 251
write-Befehl, 118, 251

X

x-Befehl (sed), 265
X Consortium, xi
xargs-Befehl, 119
xit-Befehl (ex), 251
xmkmf-Werkzeug (imake), 330-331

Y

y-Befehl (sed), 265
yacc-Befehl (Linux), 345
yank-Befehl (ex), 252
ypbind-Befehl (NFS/NIS), 441
ypcat-Befehl (NFS/NIS), 441
ypchfn-Befehl (NFS/NIS), 120
ypchsh-Befehl (NFS/NIS), 120
ypinit-Befehl (NFS/NIS), 441
ypmatch (NFS/NIS), 442
yppasswd-Befehl (NFS/NIS), 120, 442
yppoll-Befehl (NFS/NIS), 443
yppush-Befehl (NFS/NIS), 443
ypserv-Befehl (NFS/NIS), 443
ypset-Befehl (NFS/NIS), 444
ypwhich-Befehl (NFS/NIS), 444
ypxfr-Befehl (NFS/NIS), 445

Z

z-Befehl (ex), 252
Zählen von Elementen in Textdateien, 117
Zahlen
  Arithmetische Ausdrücke in der bash, 140-141
  Ausdrücke auswerten, 33
  bc Programmiersprache für, 5-9
  Oktal, für Zugriffsrechte, 13
zcat-Befehl, 121
zcmp-Befehl, 121
zdiff-Befehl, 121
zdump-Befehl, 445
Zeichen
  besondere Zeichen im Emacs-Shell-Modus, 218
  in einer Datei zählen, 117
  Metazeichen (siehe Metazeichen)
  spezielle (siehe Sonderzeichen)
  Tastaturwiederholrate, 390
  Unterstreichen, 114
Zeichenketten, Umformen von Zeichen zwischen, 112
Zeilen, Text-
  Ausgeben, 102
  Drucken, 54
  in Spalten zusammenführen, 83
  Suchen nach, 31, 51, 69
  Umbrechen, 42
  Vergleichen, 16, 26-29
Zeilenadressen (ex), 244
Zentrieren, Emacs-Befehle zum, 219
Zerteilen von Dateien, 21, 96
zforce-Befehl, 121
zgrep-Befehl, 121
zic-Befehl, 446
zmore-Befehl, 121
znew-Befehl, 122
Zugriffsrechte, Datei
  als Oktalzahl, 13
  Eigentümer, 13
  Gruppen, 11, 83
  Ändern, 12
Zwischenablage
  Emacs, 216

Über die Autorin

Jessica Perry Hekman lebt in Somerville, Massachusetts, mit wechselnden Mitbewohnern, zwei prächtigen Katzen und Hunderten von Büchern. Nachdem Sie bei O'Reilly & Associates bereits als Herstellungsassistentin, Systemverwalterin und Autorin tätig war, arbeitet sie heute als Entwicklerin im Bereich der Online-Dienste. Sie schloß 1995 ihr Studium an der Harvard University mit *magna cum laude* ab; ihr Fachgebiet war die Geschichte und Literatur des Mittelalters unter besonderer Berücksichtigung von Wales und Frankreich. Ihre Lieblingsfarbe ist grün, ihre Lieblingsautorin Ursula K. LeGuin und ihr persönlicher Held J. Michael Straczynski.

Über den Übersetzer

Matthias Kalle Dalheimer arbeitet als freier Autor, Übersetzer, Gutachter und Software-Entwickler in Hamburg, wo er mit seiner Frau und seinem Sohn Jan Lennart zusammenlebt. Nach dem Studium der Informatik und Allgemeinen Sprachwissenschaft arbeitete er zunächst bei Star Division und war dort maßgeblich an der Portierung des StarOffice auf Linux beteiligt. Im Frühjahr 1997 gab er seiner Liebe zu Büchern nach und wechselte die Branche. Kalle arbeitet unter Linux und benutzt XEmacs 19.15 für den Großteil seiner Arbeit. In seiner Freizeit spielt er mit seinem Sohn, wandert oder liest Geschichtsbücher.

Kolophon

Das Erscheinungsbild unserer Bücher ist das Ergebnis von Leserkommentaren, eigenen Erfahrungen und Feedback aus den Vertriebskanälen. Die unverwechselbaren Covergestaltungen widerspiegeln unsere besondere Herangehensweise an technische Themen, die potentiell trockenen Inhalten Leben einhaucht.

Das Tier auf dem Buchtitel von *Linux in a Nutshell* ist ein Araber. Diese Pferderasse, die für ihre Anmut und Intelligenz bekannt ist, ist eine der ältesten überhaupt; die ältesten Beweise für ihre Existenz sind 5000 Jahre alt. Die Rasse der Araberpferde bildete auch den Ausgangspunkt für eine Reihe bekannter Züchtungen wie z.B. der Vollblutpferde des 17. und 18. Jahrhunderts. Sie gilt als eine der besonders charakteristischen Rassen; die Tiere zeichnen sich durch große, ausdrucksvolle Augen und Nüstern, kleine Ohren und einen kurzen, stämmigen Rücken aus. Wegen ihrer Zähigkeit eignen sich Araber besonders für Distanzritte, eine Sportart, in der sie vorherrschend sind. Ihr wundervolles Temperament hat die Araber in Nordamerika zu einer beliebten Reitpferderasse gemacht, obwohl sie auch in spezialisierteren Sparten wie Dressur oder Springreiten erfolgreich sind.

Edie Freedman gestaltete den Umschlag dieses Buchs und verwendete hierfür eine Gravur aus dem Dover Pictorial Archive. Das Titellayout wurde von Hanna Dyer mit Quark XPress 3.3 und ITC Garamond erstellt. Der Text wurde von Christoph Badura mit dem Textformatierer troff gesetzt. Das Kolophon schrieb Nicole Gipson Arigo.

Deutsche Ausgabe

LINUX IN A NUTSHELL

Linux in a Nutshell beschreibt alle wichtigen Befehle für Benutzer, Programmierer und Systemadministratoren, die in den verbreitetsten Linux-Distributionen enthalten sind.

Viele Unix-Benutzer, die die GNU-Versionen der Standard-Unix-Werkzeuge verwenden, werden dieses Buch ebenfalls nützlich finden, weil es viele der GNU-Werkzeuge dokumentiert.

Linux in a Nutshell behandelt:

- Benutzer-Befehle und deren Optionen
- Die Syntax der Shells bash und csh/tcsh
- Mustervergleiche
- Die Editoren Emacs, vi und ex
- Die Skriptsprachen sed, gawk und Perl
- Programmierbefehle
- Befehle zur System- und Netzwerkadministration

Dieses Buch enthält alle notwendigen Befehle, die Sie brauchen, um Ihr System zu betreiben, und die meisten Befehle, die traditionell in Unix-Systemen enthalten sind. Spezielle Pakete der verschiedenen Linux-Distributionen werden nicht beschrieben.

Alle Themen sind in übersichtlichen Kapiteln dargestellt, in denen Sie die gewünschten Informationen in Sekundenschnelle parat haben. *Linux in a Nutshell* ist daher das Nachschlagewerk für die tägliche Arbeit mit Linux.

ISBN 3-930673-57-6
04900
9 783930 673575

DEM 49.-
ATS 358.-
ISBN 3-930673-57-6

O'REILLY™